ILLINOIS INSTITUTE OF TECHNOLOGY

Amazon Product Review Analysis & Recommendation System

CSP 571 Data Preparation and Analysis
Prof. Jawahar Panchal

A20563950
Satwika Sriram
ssriram6@hawk.iit.edu

A20552681
Naga Sunith Appasani
nappasani@hawk.iit.edu

1. ABSTRACT

In the age of e-commerce, consumer reviews play a crucial role in shaping purchasing decisions. This project aims to develop an Amazon Product Review Analysis and Recommendation System to enhance the shopping experience by providing insightful analysis and tailored recommendations. Leveraging advanced natural language processing (NLP) and machine learning techniques, the system extracts and interprets user reviews to gauge product sentiment, identify key features, and detect trends. The core components include sentiment analysis to classify reviews into positive, negative, or neutral categories, topic modelling to uncover prevalent themes, and a recommendation engine that suggests products based on user preferences and historical data. By integrating these components, the system not only aids consumers in making informed decisions but also offers sellers actionable insights into customer feedback. The result is a robust platform that enhances the e-commerce experience through data-driven insights and personalised recommendations, potentially leading to improved customer satisfaction and optimised product offerings.

2. INTRODUCTION

This project centres on the development of an Amazon Product Review Analysis and Recommendation System, utilising a rich dataset sourced from Kaggle. This dataset encompasses over 1500 Amazon products, detailing their names, categories, prices, ratings, and user reviews. Our objective is to construct a comprehensive system that not only analyses product reviews but also offers personalised recommendations to enhance the shopping experience.

The project is structured through several key stages:

1. **Data Preparation**: The initial stage involves collecting the dataset and addressing any missing or inconsistent information to ensure high data quality. This preparation is essential for the subsequent stages of analysis.
2. **Exploratory Data Analysis (EDA)**: We conduct EDA to explore the dataset's structure and characteristics. This phase includes examining the distribution of reviews, uncovering key features, and identifying patterns or trends that may impact the analysis.
3. **Data Pre-processing**: Text data undergoes cleaning, tokenization, and normalisation to standardise the input for sentiment analysis. This pre-processing step is crucial for minimising noise and ensuring that the data is in a consistent format.
4. **Sentiment Labelling**: Reviews are categorised into sentiment classes—positive, negative, or neutral—based on the expressed attitudes. This labelling process sets the foundation for detailed sentiment analysis and model training.
5. **Text Vectorization**: To facilitate machine learning, textual data is converted into numerical formats using techniques such as TF-IDF vectorization or word embeddings. This transformation allows algorithms to process and analyse the text effectively.
6. **Model Development**: We experiment with various machine learning models, including k-nearest neighbors, random forest, support vector machines, and neural networks, to

classify sentiment. Each model is trained on the labelled dataset to assess its ability to predict sentiment accurately.

7. **Model Evaluation**: The performance of each model is evaluated using metrics such as accuracy, precision, recall, and F1-score. This evaluation helps determine the most effective approach for sentiment classification.
8. **Predictive Modelling:** Developing and evaluating a Random Forest Regression model to predict discount percentages and rating based on product attributes such as actual price and category.
9. **Recommendation System:** Developing a recommendation system leveraging machine learning algorithms to suggest products tailored to users' interests and prior purchases based on their past review and ratings.

3. PROPOSED METHODOLOGY

By implementing this system, we aim to demonstrate the power of sentiment analysis in deriving meaningful insights from Amazon reviews and showcase how data science can significantly improve the understanding of customer sentiments in the digital marketplace. This project not only provides a valuable tool for consumers but also offers businesses a strategic advantage in navigating the competitive e-commerce landscape.

Collected a comprehensive dataset from Kaggle and processed it by cleaning, normalising, and tokenizing to ensure data quality. Conducted exploratory data analysis to uncover patterns and insights in the review data. Utilised text vectorization methods to prepare the data for machine learning. Developed and refined various sentiment classification models, evaluating their effectiveness using metrics such as accuracy and F1-score. Developed a recommendation system to suggest products to users based on their past purchases and history.

4. ASSUMPTIONS

We assume that the product review dataset is a true reflection of the target population, capturing a diverse array of sentiments. It is expected that the sentiment labels assigned to each review accurately represent the sentiments conveyed in the text. Additionally, we presume that the text preparation methods effectively remove noise and irrelevant information, thereby improving the quality of the sentiment analysis. We anticipate that the machine learning models, trained on the labelled dataset, will generalise well to new data and produce reliable sentiment predictions. Lastly, we develop a recommendation system to suggest products to users based on their past purchases and history.

5. DATA EXPLORATION

5.1 DATA DESCRIPTION

The dataset, "amazon.csv," used in this project contains attributes of Amazon product reviews sourced from Kaggle, comprising 1465 rows and 16 columns. It encompasses a wide range of

product reviews characteristics such as product_name, category, discounted_price, actual_price, rating, user_id, user_name, review content and other product attributes. These features are utilised to predict products to users based on their interests and previous purchases.

Data Source:

https://www.kaggle.com/datasets/karkavelrajaj/amazon- sales-dataset

```
  product_id        product_name         category        discounted_price   actual_price       discount_percentage    rating
 Length:1465        Length:1465        Length:1465        Length:1465        Length:1465        Length:1465          Length:1465
 Class :character   Class :character   Class :character   Class :character   Class :character   Class :character     Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character     Mode  :character
 rating_count       about_product        user_id            user_name          review_id          review_title       review_content
 Length:1465        Length:1465        Length:1465        Length:1465        Length:1465        Length:1465          Length:1465
 Class :character   Class :character   Class :character   Class :character   Class :character   Class :character     Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character     Mode  :character
   img_link         product_link
 Length:1465        Length:1465
 Class :character   Class :character
 Mode  :character   Mode  :character
```

Fig: Above depicts the summary of the Dataset

```
 [1] "product_id"          "product_name"        "category"          "discounted_price"    "actual_price"        "discount_percentage"
 [7] "rating"              "rating_count"        "about_product"     "user_id"             "user_name"           "review_id"
[13] "review_title"        "review_content"
```

Fig: Above depicts the columns in the Dataset.

```
Rows: 5
Columns: 14
$ product_id          <chr> "B07JW9H4J1", "B098NS6PVG", "B096MSW6CT", "B08HDJ86NZ", "B08CF3B7N1"
$ product_name        <chr> "Wayona Nylon Braided USB to Lightning Fast Charging and Data Sync Cable Compatible for iPhone 13, 12,11, X, 8, 7, 6, 5, iPad Air, …
$ category            <chr> "Computers&Accessories|Accessories&Peripherals|Cables&Accessories|Cables|USBCables", "Computers&Accessories|Accessories&Peripherals…
$ discounted_price    <chr> "₹399", "₹199", "₹199", "₹329", "₹154"
$ actual_price        <chr> "₹1,099", "₹349", "₹1,899", "₹699", "₹399"
$ discount_percentage <chr> "64%", "43%", "90%", "53%", "61%"
$ rating              <chr> "4.2", "4", "3.9", "4.2", "4.2"
$ rating_count        <chr> "24,269", "43,994", "7,928", "94,363", "16,905"
$ about_product       <chr> "High Compatibility : Compatible With iPhone 12, 11, X/XsMax/Xr ,iPhone 8/8 Plus,iPhone 7/7 Plus,iPhone 6s/6s Plus,iPhone 6/6 Plus,…
$ user_id             <chr> "AG3D6O4STAQKAY2UVGEUV46KN35Q,AHMY5CWJMMK5BJRBBSNLYT3ONILA,AHCTC6ULH4XB6YHDY6PCH2R772LQ,AGYHHIERNXKA6P5T7CZLXKVPT7IQ,AG4OGOFWXJZTQ2…
$ user_name           <chr> "Manav,Adarsh gupta,Sundeep,S.Sayeed Ahmed,jaspreet singh,Khaja moin,Anand,S.ARUMUGAM", "ArdKn,Nirbhay kumar,Sagar Viswanathan,Asp,…
$ review_id           <chr> "R3HXWT0LRP0NMF,R2AJM3LFTLZHFO,R6AQJGUP6P86,R1KD19VHEDV0OR,R3C02RMYQMK6FC,R39GQRVBUZBWGY,R2K9EDOE15QIRJ,R3OI7YT648TL8I", "RGIQEG07R…
$ review_title        <chr> "Satisfied,Charging is really fast,Value for money,Product review,Good quality,Good product,Good Product,As of now seems good", "A …
$ review_content      <chr> "Looks durable Charging is fine tooNo complains,Charging is really fast, good product.,Till now satisfied with the quality.,This is…
```

Fig: Above depicts the top few rows in the Dataset

5.2 DATA PREPROCESSING, CLEANING AND WRANGLING

In the initial phase of the project, we carefully sourced a dataset from Kaggle, which comprised thousands of Amazon product reviews. This dataset presented notable challenges due to its raw and disorganised nature. To address these issues, we conducted a comprehensive data cleaning process to improve accuracy and consistency. The cleaned data was then organised into a Pandas Data Frame, facilitating efficient manipulation and analysis. This preparation was essential for advancing to data exploration and setting the foundation for effective sentiment analysis.

5.2.1 Data Inspection

Data inspection is an essential preliminary step in preparing a dataset for analysis, aimed at ensuring data quality and integrity. The first aspect of this process involves identifying missing values within the dataset. This is accomplished by examining the dataset for 'NaN' (Not a Number), 'Null', or similar indicators of missing data. Summary statistics and visual tools such as heat maps are employed to understand the extent and distribution of missing values. Evaluating the proportion of missing data helps determine whether it is substantial and whether imputation, deletion, or other strategies are needed to address these gaps.

product_id	product_name	category	discounted_price	actual_price	discount_percentage	rating
0	0	0	0	0	0	0
rating_count	**about_product**	**user_id**	**user_name**	**review_id**	**review_title**	**review_content**
2	0	0	0	0	0	0

Fig: Above depicts the missing values in columns of Dataset

To ensure the accuracy of our analysis, we'll remove rows from the dataset that contain blank values for the rating_count column. This step is crucial as missing information can impact the reliability of our findings. By doing so, we'll focus on preserving data points that have all the necessary details intact.

product_id	product_name	category	discounted_price	actual_price	discount_percentage	rating
0	0	0	0	0	0	0
rating_count	**about_product**	**user_id**	**user_name**	**review_id**	**review_title**	**review_content**
0	0	0	0	0	0	0

Fig: Above depicts the missing values removed in rating count column

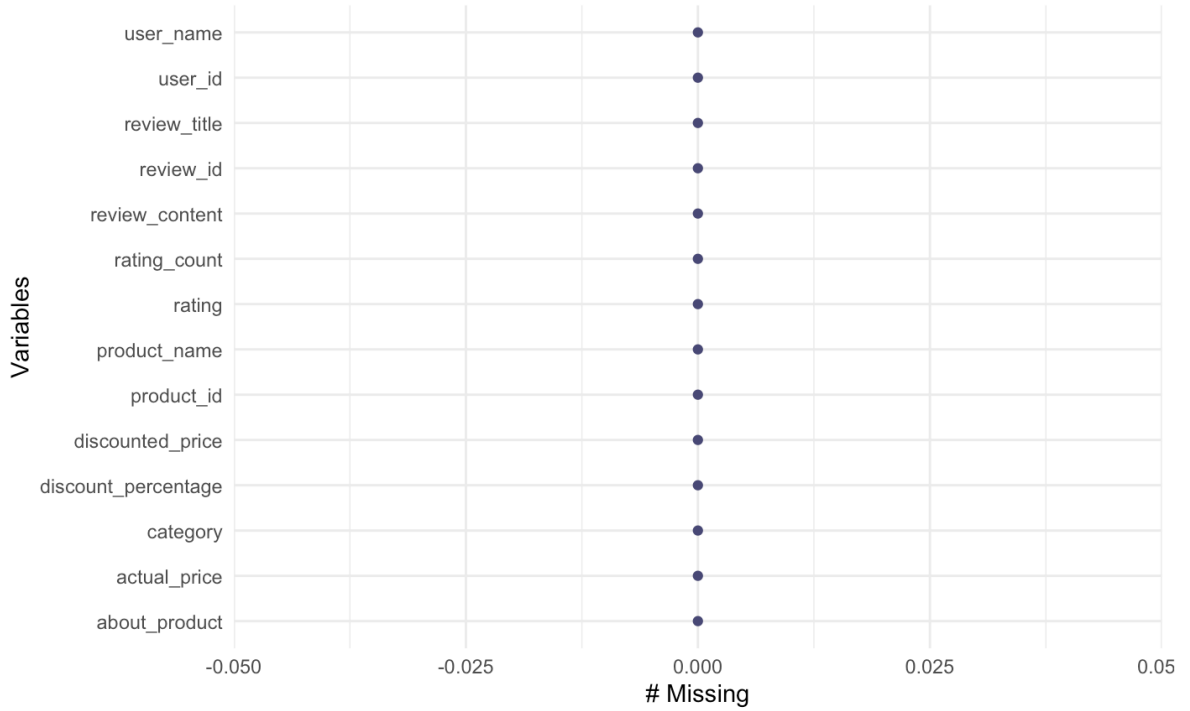


Fig: Above heatmaps depicts distribution of missing values

The next aspect of this process is to identify and resolve any duplicate entries to prevent skewed analysis.

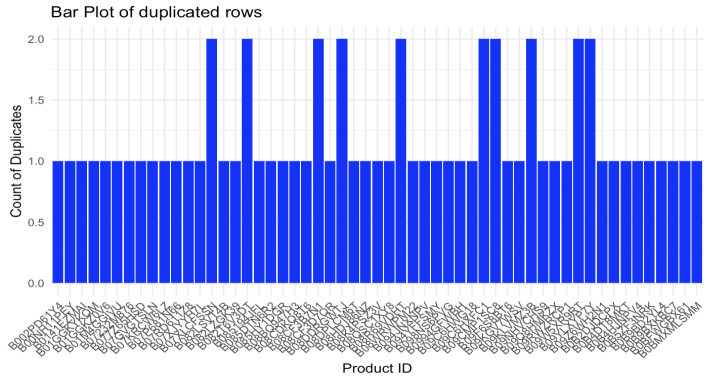


Fig: Above Bar plot depicts distribution of duplicate values.

After eliminating all the duplicate values from the dataframe, we move on to data transformation.

```r
# Check for duplicate rows in the dataframe
duplicates <- duplicated(df)

# Subset the dataframe to include only duplicated rows
duplicated_rows <- df[duplicates, ]

# Print or inspect the duplicated rows
print(duplicated_rows)
```

Description: df [0 × 16]

0 rows | 1–7 of 16 columns

We observe that there are no duplicate values in our dataframe.

5.2.2 Data Transformation

Upon data cleaning and standardisation steps, we proceeded with the data transformation phase to further optimise our dataset for analysis. This phase involved refining and categorising the data to align it with our analytical goals. **Encoding categorical variables** step, involves categorical data is converted into numerical formats suitable for algorithms. **Normalisation and scaling** is another crucial step, where we adjust the range of numerical data so that all features contribute equally to the analysis. This process often uses methods like Min-Max scaling, which rescales values to a specified range, or Z-score normalisation, which standardised data based on mean and standard deviation. This can be done through methods such as one-hot encoding, which creates binary columns for each category, or label encoding, which assigns numerical values to categories. Additionally, **feature engineering** involves creating new features or refining existing ones to enhance the dataset's relevance and effectiveness for the intended analysis or model training. This process helps improve the overall performance and interpretability of the analysis or predictive models.

product_id	product_name	category	discounted_price	actual_price	discount_percentage	rating
"character"	"character"	"character"	"character"	"character"	"character"	"character"
rating_count	about_product	user_id	user_name	review_id	review_title	review_content
"character"	"character"	"character"	"character"	"character"	"character"	"character"

Fig: Above depicts data types of columns in data frame.

Converting price, rating columns from character to numeric and rating column has inconsistent value and is replaced with the highest rating. Further checked for missing values after conversion of datatypes.

product_id	product_name	category	discounted_price	actual_price	discount_percentage	rating
"character"	"character"	"character"	"numeric"	"numeric"	"numeric"	"numeric"
rating_count	about_product	user_id	user_name	review_id	review_title	review_content
"numeric"	"character"	"character"	"character"	"character"	"character"	"character"

Fig: Above depicts data types of columns after transformation in data frame.

Further, we focused on refining the dataset by implementing a series of transformations. First, the 'category' column was divided into two distinct columns: 'main_category' and 'sub-category'. This division improved the organisation and granularity of the data. This extraction was achieved using the ‘gsub’ and ‘substr’ functions.

Next, we processed several columns - namely 'review_id', 'review_title', 'review_content', and 'user_id' by splitting them according to specific delimiters. This step was crucial for breaking down complex data into more manageable parts, facilitating better analysis. This extraction was achieved using the ‘sapply’ and ‘strsplit’ functions.

To provide a more descriptive understanding of ratings, a new column ‘rating_score_values’ was created based on numerical rating values. The objective was to transform numeric ratings into qualitative categories that better reflect performance levels.

Finally, after the columns were split, we reintegrated the newly created columns back into the original dataframe using the ‘cbind’ function. This ensured that the data frame remained complete and ready for further analysis, enhancing its overall usability.

```
      product_id       product_name           category   discounted_price
     "character"        "character"        "character"          "numeric"
   actual_price discount_percentage             rating       rating_count
      "numeric"          "numeric"          "numeric"          "numeric"
  about_product            user_id          user_name          review_id
     "character"        "character"        "character"        "character"
   review_title     review_content      main_category       sub_category
     "character"        "character"        "character"        "character"
    review_id_1        review_id_2        review_id_3        review_id_4
     "character"        "character"        "character"        "character"
    review_id_5        review_id_6        review_id_7        review_id_8
     "character"        "character"        "character"        "character"
  review_title_1     review_title_2     review_title_3     review_title_4
     "character"        "character"        "character"        "character"
  review_title_5     review_title_6     review_title_7     review_title_8
     "character"        "character"        "character"        "character"
review_content_1   review_content_2   review_content_3   review_content_4
     "character"        "character"        "character"        "character"
review_content_5   review_content_6   review_content_7   review_content_8
     "character"        "character"        "character"        "character"
      user_id_1          user_id_2          user_id_3          user_id_4
     "character"        "character"        "character"        "character"
      user_id_5          user_id_6          user_id_7          user_id_8
     "character"        "character"        "character"        "character"
rating_score_values
     "character"
```

Fig: Above depicts dataset after all data preparations.

6. EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) is an essential step where we begin to investigate the dataset to identify patterns, spot anomalies, test hypotheses, and confirm assumptions. By employing summary statistics and visual tools, EDA provides a comprehensive view of the data's underlying patterns and characteristics. This phase helps us grasp the intricate relationships between the response variable and predictor variables, offering key insights into the elements that impact the dataset. Such exploration lays the foundation for more detailed analyses and the creation of predictive models.

Summarise numerical variables:

We computed summary statistics for numerical columns of the dataset - 'discounted_price', 'actual_price', 'discount_percentage', 'rating', 'rating_count' including the mean, median, minimum, maximum, and standard deviation, for numerical variables to gain insights into their distribution and central tendencies. This provides insights into pricing strategies, discounting practices, and customer satisfaction in our project.

	discounted_price	actual_price	discount_percentage	rating	rating_count
Min.	39.000	39.000	0.0000000	2.000000	2.00
1st Qu.	347.250	899.000	0.3100000	4.000000	1183.50
Median	861.500	1699.500	0.5000000	4.100000	5104.50
Mean	3234.072	5620.271	0.4722389	4.093419	18509.38
3rd Qu.	2098.000	4637.500	0.6300000	4.300000	17414.50
Max.	77990.000	139900.000	0.9400000	5.000000	426973.00

Fig: Above depicts statistical summary of numerical columns

The data reveals diverse pricing trends, with a broad spectrum of product prices. The average discounted price is notably higher than the median, indicating that while many products are priced lower, high-end items are inflating the average. Discounts are quite significant overall, with an average discount of 47% and some reaching as high as 94%, reflecting aggressive pricing strategies. Additionally, products generally receive high average ratings, pointing to overall customer satisfaction. However, there is considerable variability in the number of ratings, with some products attracting significantly more feedback than others.

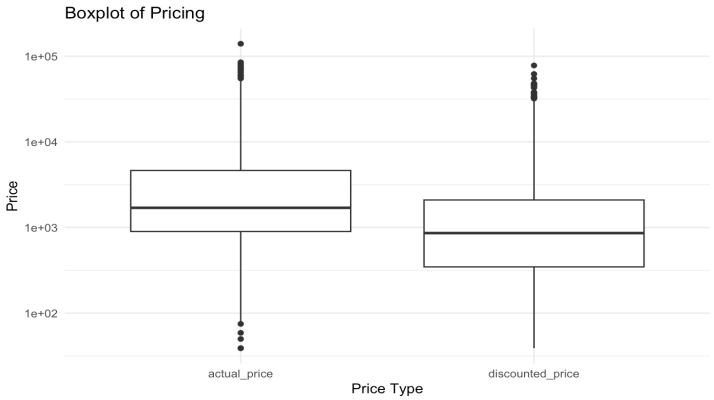


Fig: Above depicts box plot of price

We observe that actual prices are more variable and higher on average compared to the discounted prices.

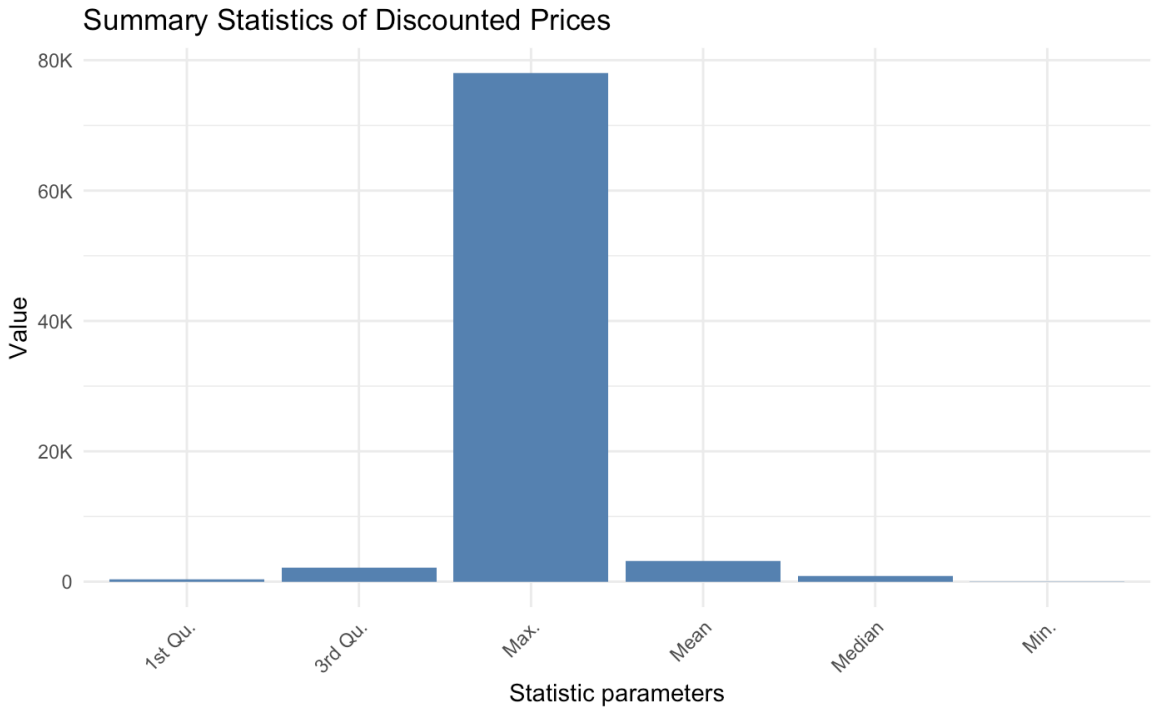


Fig: Above depicts bar plot for discounted prices

We observe most discounted prices are in the lower to mid-range, a few very high prices are significantly influencing the average. The data is right-skewed i.e., discounted prices are relatively lower, there are a few higher prices that raise the mean significantly.

Distribution of Products by rating:

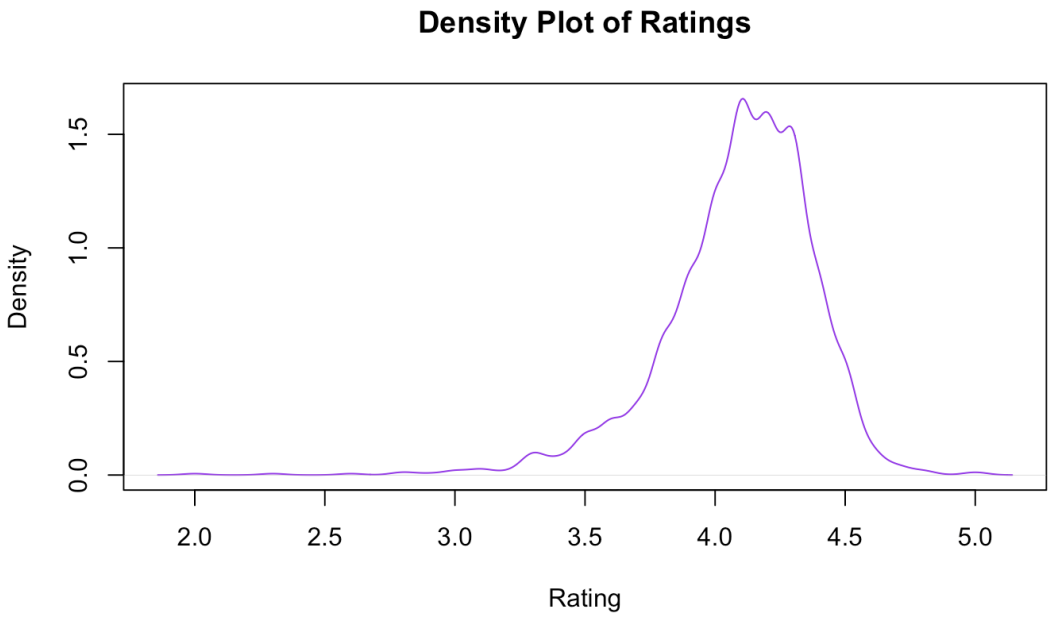


Fig: Above depicts density plot of ratings

It is observed that the density plot indicates the distribution of ratings, highlighting that most items receive high ratings with some variability. Majority of ratings are concentrated around a central value, specifically near 4.1. This suggests that most items are rated relatively highly, with 4.1 being a common rating.

Distribution of Products by category:

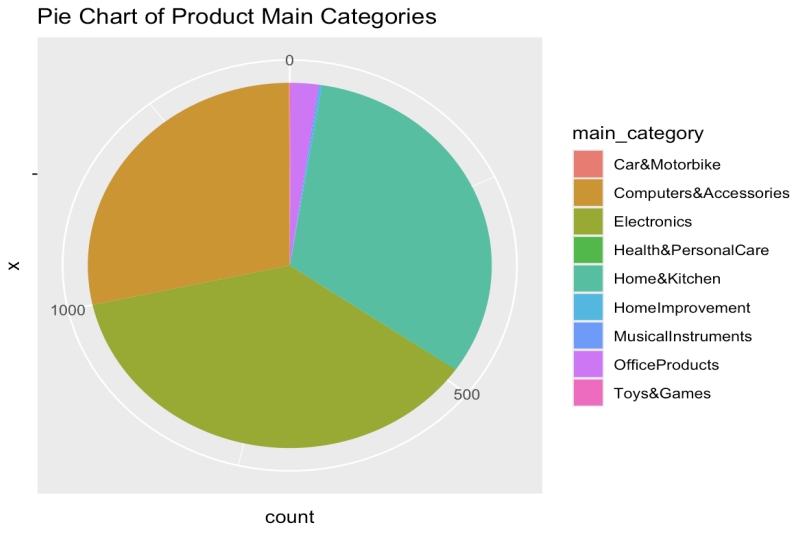


Fig: Above depicts pie chart of main categories

The data reveals that **Electronics** (515) and **Home & Kitchen** (448) are the most prominent categories, indicating their high demand and broad applicability. **Computers & Accessories** (397) also holds a significant share, though it is slightly less prominent than the top two categories. **Office Products** (31) falls into a moderate range, representing a noteworthy but smaller portion compared to the top categories. In contrast, categories such as **Car & Motorbike**, **Health & Personal Care**, and **Toys & Games** have very low counts (1), suggesting they may be niche markets or have limited data representation. **Home Improvement** and **Musical Instruments**, with counts of 2, are slightly more represented but still relatively minor. These insights suggest a strong focus on Electronics and Home & Kitchen, while also highlighting potential areas for further investigation into less represented categories.

Distribution of Products by sub-category:

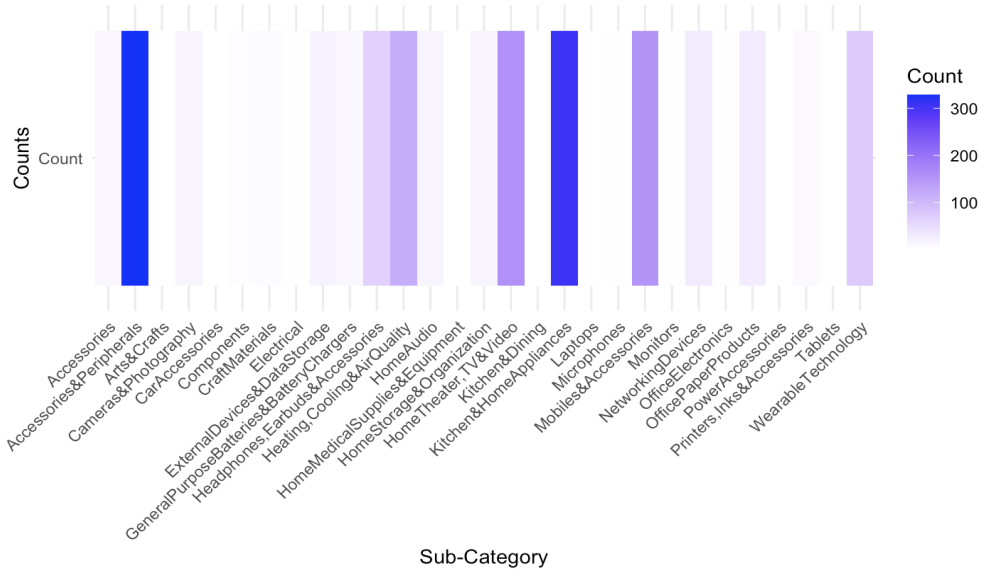


Fig: Above depicts map of subcategories

The data reveals a varied distribution of products across categories. Major subcategories such as **Accessories & Peripherals**, **HomeTheater, TV & Video**, and **Kitchen & Home Appliances** feature the highest quantities, indicating a strong focus and extensive inventory in these areas. **Headphones, Earbuds & Accessories** and **Heating, Cooling & Air Quality** also have significant counts, reflecting a notable range of products. Conversely, several categories like **Arts & Crafts**, **Car Accessories**, and **Tablets** have very few items, suggesting potential niche markets or areas for inventory expansion. The data suggests that while there is a clear emphasis on certain product lines, there may be opportunities to grow or refine offerings in categories with limited items.

Distribution of expensive products after discount:

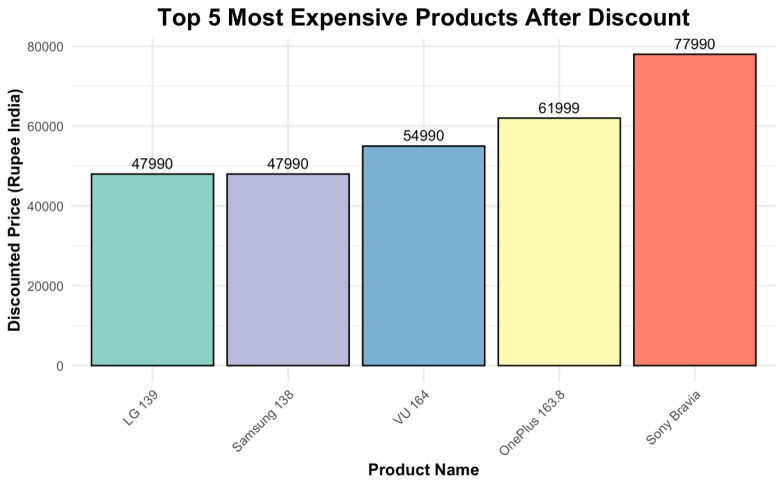


Fig: Above depicts bar plot of product vs price of top 5 expensive products.

It can be observed that electronics are expensive and most discounted.

Distribution of cheapest products after discount:

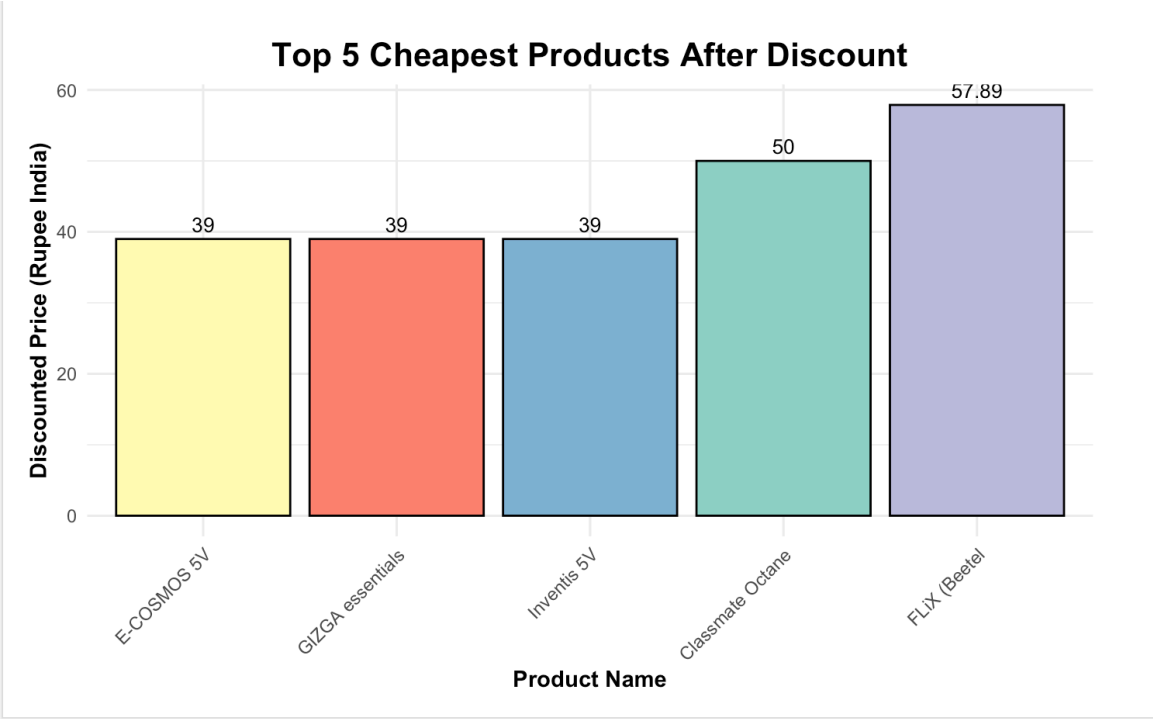


Fig: Above depicts bar plot of product vs price of top 5 least expensive products.

The bar chart shows the top 5 cheapest products after discount, with E-COSMOS 5V, GIZGA essentials, and Invertis 5V all priced at 39 Rupees, while FLX (Beetel) is the most expensive among the cheapest products at 57.89 Rupees, highlighting the affordable options available.

Correlation between features:

We further investigated correlations of most relationships between these variables using Heatmap.

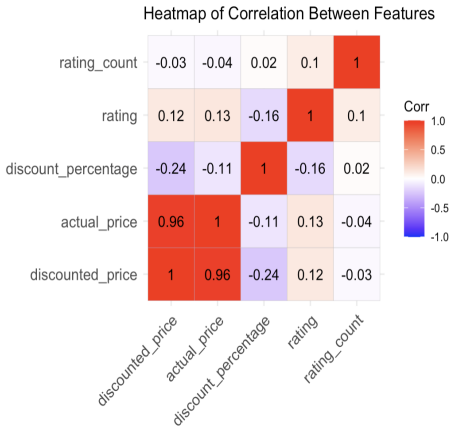


Fig: Correlation matrix for features

The correlation matrix reveals that the discounted price and actual price have a strong positive relationship, indicating they tend to increase together. The other variables show mostly weak correlations: discount percentage has a weak negative relationship with both the discounted price and actual price; rating and rating count have very weak correlations with all other variables, suggesting minimal impact. Overall, the data suggests that while the discounted and actual prices are closely linked, other relationships among the variables are weak and do not show strong dependencies.

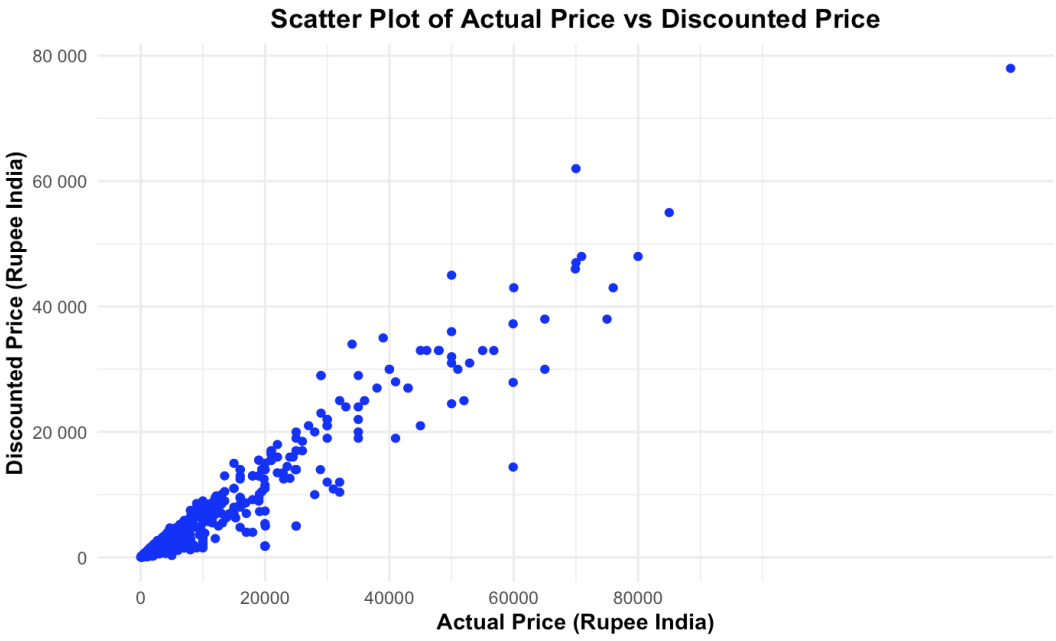


Fig: Scatter plot for price vs discount

We can observe that the higher the price of the product, the greater the discount.

Distribution of discount percentage range by main category:

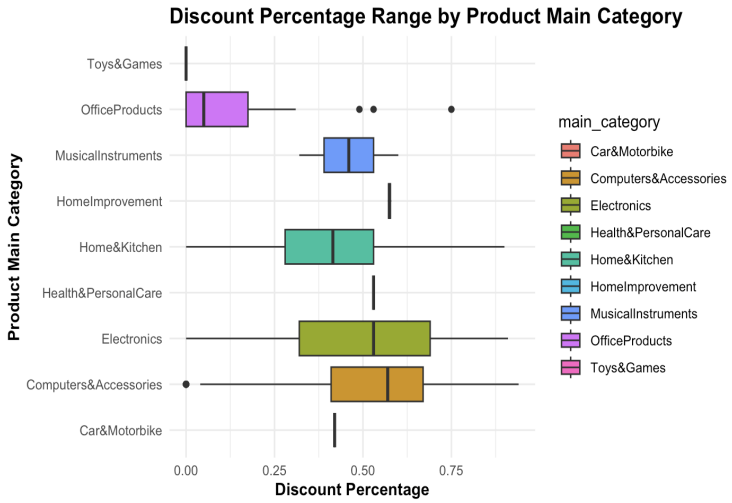
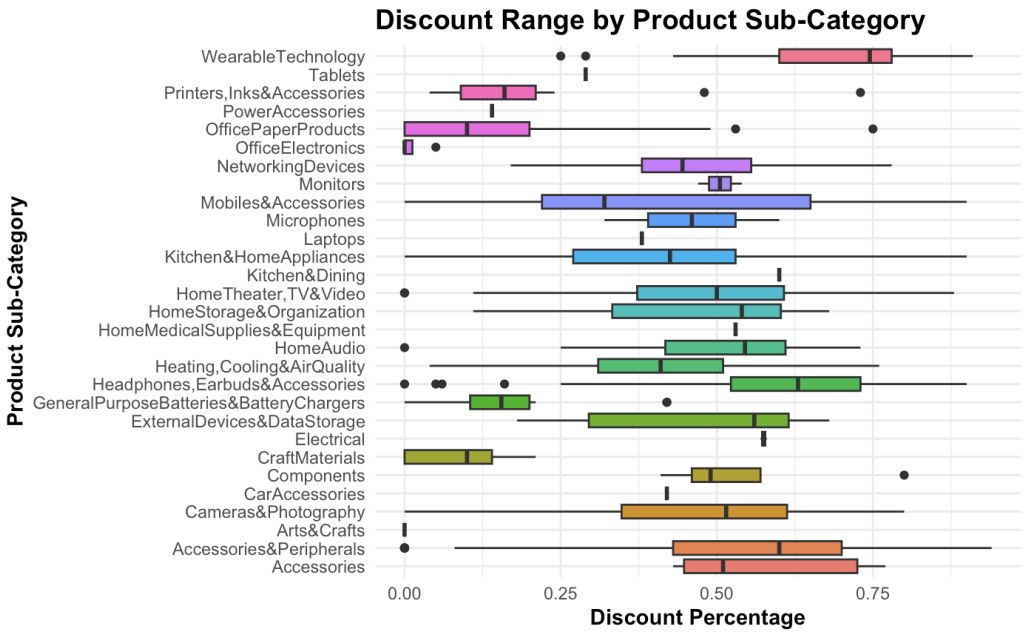


Fig: Plot of discount over categories

Computers & Accessories, Electronics, and Home & Kitchen items exhibit a wide range of discounts, from 0% up to over 90%. On the other hand, Toys & Games, Cars & Motorbikes, Health & Personal Care, and Home Improvement products have the narrowest range of discount variations.

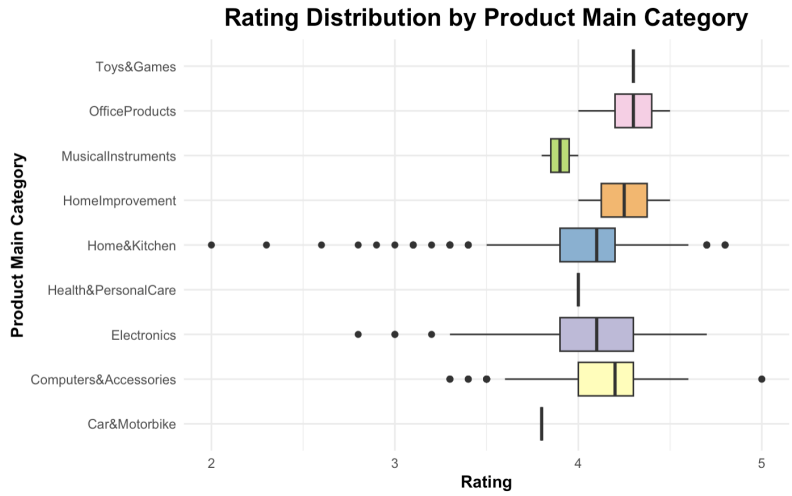
Distribution of discount percentage range by sub-category:



We observe that wearable technology products have the highest discount percentage and office electronics have the list. This suggests that these categories may have some level of price sensitivity, but not to the same extent as HomeImprovement, Computers & Accessories, and

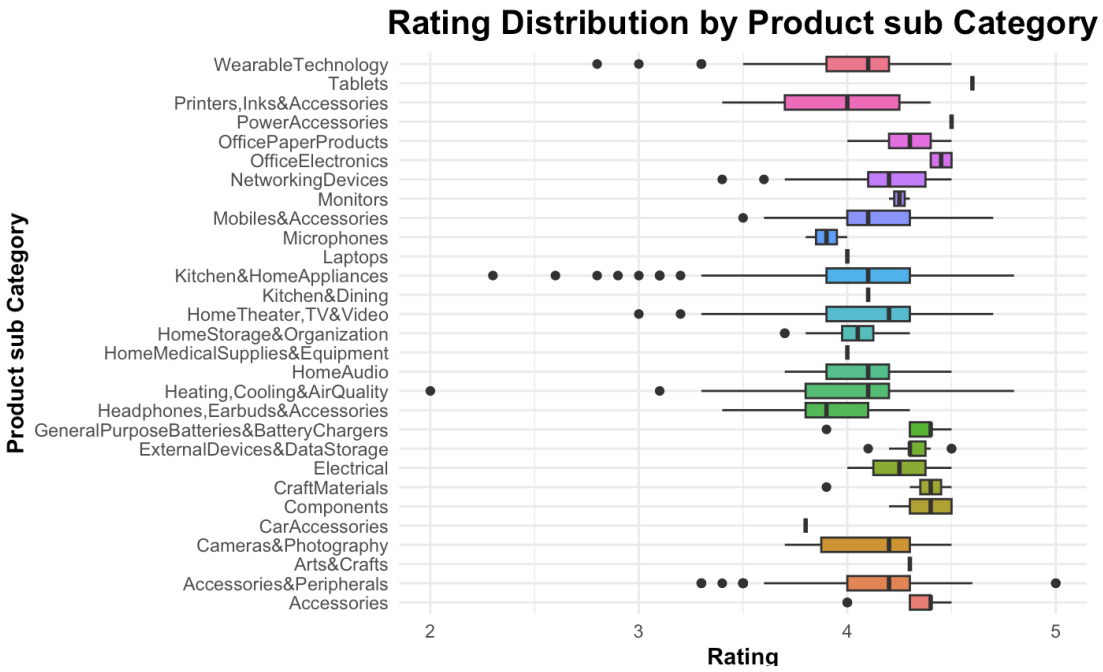
Electronics. This also provides very useful insights to recommend products to users who are more interested in higher discount percentage.

Distribution of rating for all the products by category:



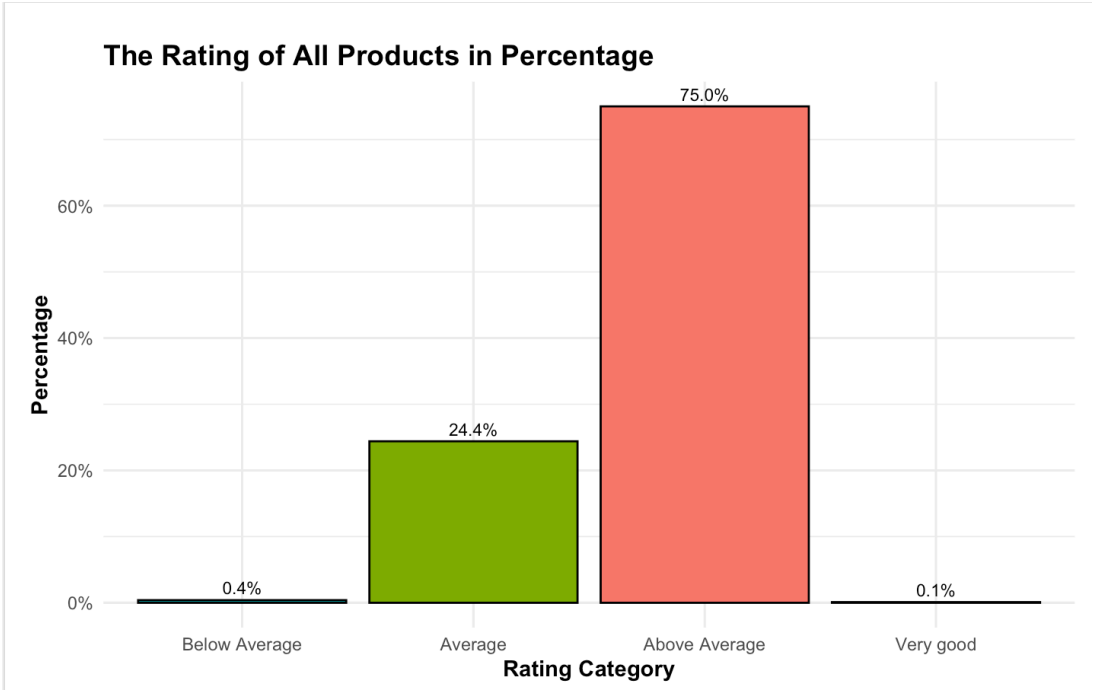
Above box plot provides useful insight to shortlist categories for our sentiment analysis. Electronics have higher ratings and reviews.

Distribution of rating for all the products by sub-category:



This helps us further deep analysis for subcategory levels of products.

Distribution of rating score values:



The bar chart illustrates the distribution of product ratings, showing that 75% of all products fall into the "Above Average" category, indicating a generally high level of customer satisfaction. In contrast, only 0.4% and 0.1% of products are rated as "Below Average" and "Very Good," respectively, while 24.4% of products are rated as "Average." This distribution suggests that most products meet or exceed customer expectations, with a significant majority being rated favourably.

Distribution of Sentiments

Before understanding the sentiments of reviews, let's look at word frequency analysis to pinpoint the most used words and phrases in the reviews. This allowed us to understand the language that customers frequently use in their feedback. Word clouds are created to visualise the most frequently occurring words in the dataset. This provided a more intuitive representation of the prominent terms used in the reviews.

The word clouds illustrate the most frequently mentioned words in product reviews. The top word cloud, representing the entire dataset, highlights key terms such as "quality," "price," and "use," indicating common themes in customer feedback. The bottom word cloud differentiates between products rated above and below average, with words like "quality" prominently appearing in both, suggesting these factors are critical in customer evaluations.

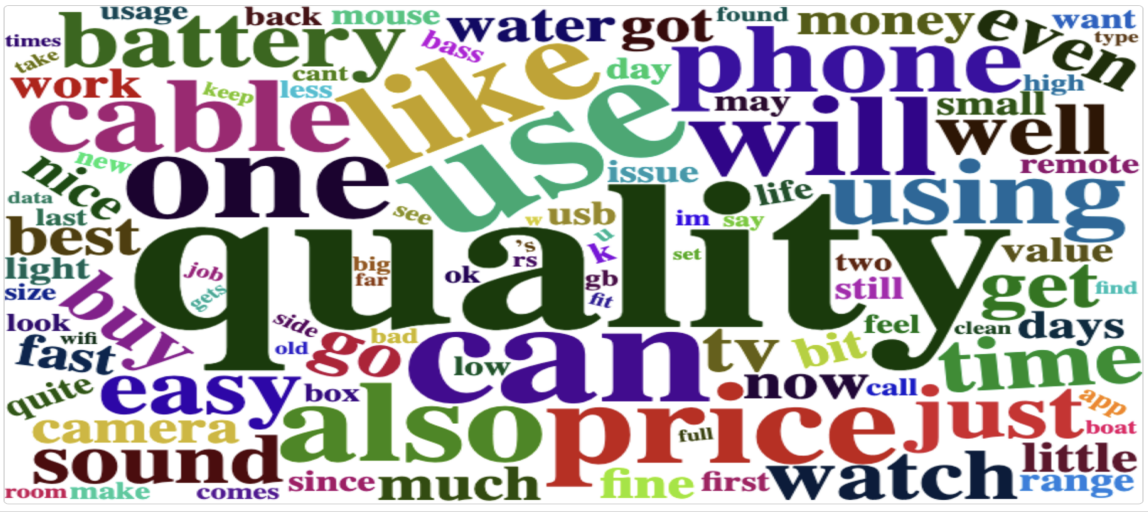


Fig: Above word cloud is over entire review of products

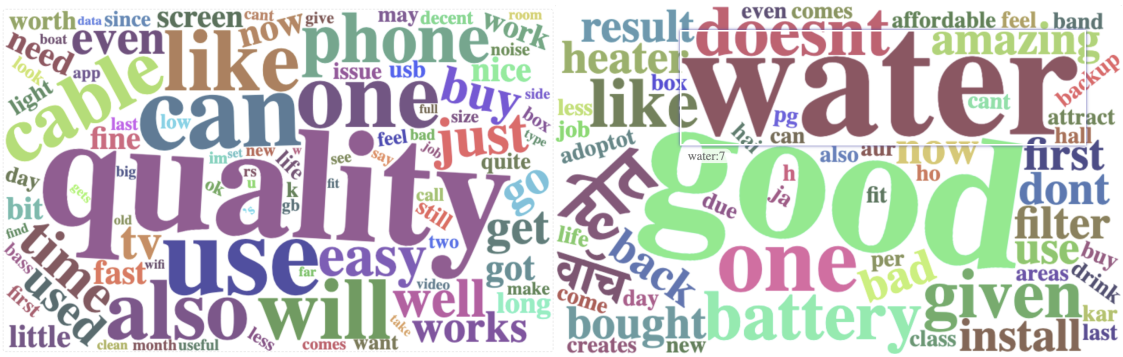
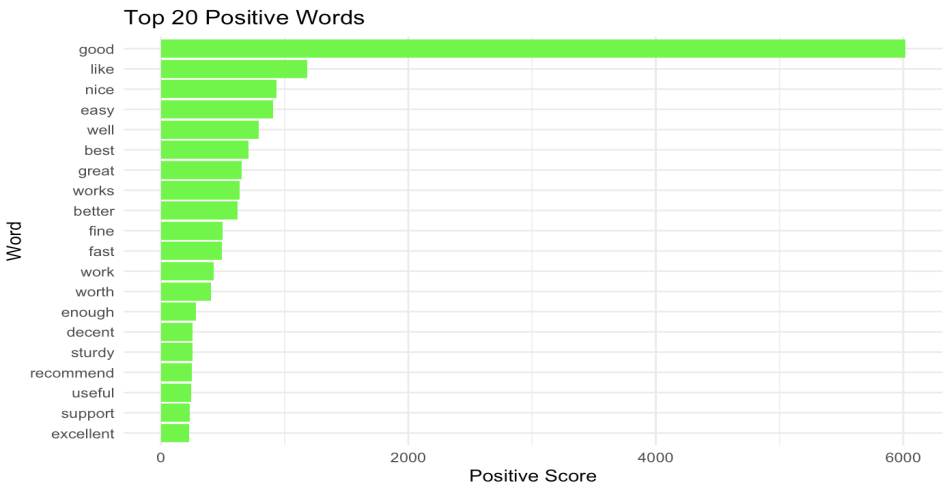


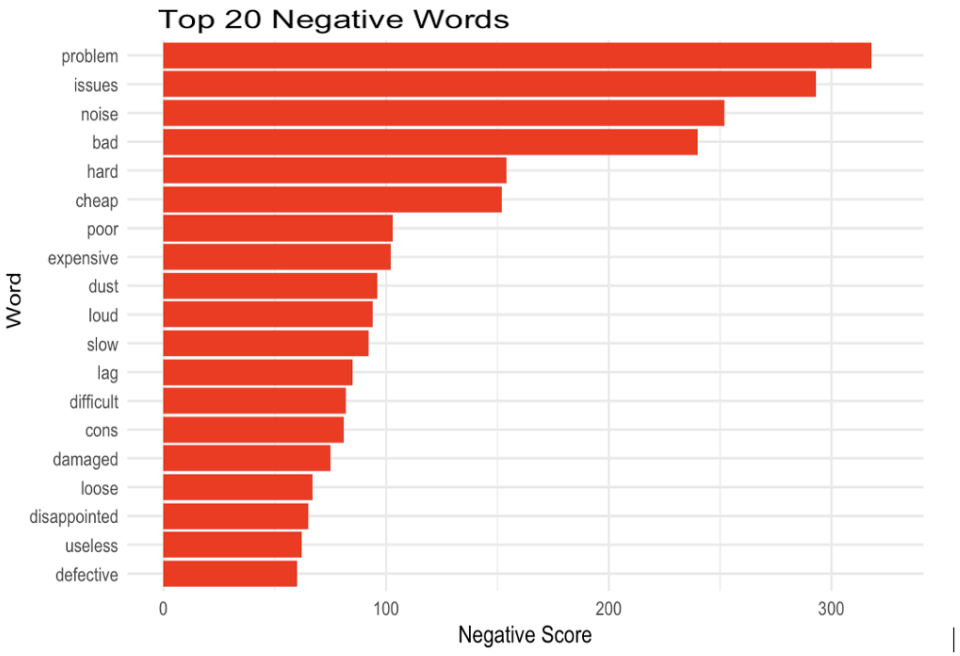
Fig: Above word clouds distribution of products with rating above and below average

To gain insights into the sentiment characteristics of our dataset that is required for our prediction engine, we visualised the distribution of sentiments—both positive and negative. This visualisation helps us understand the overall sentiment trends and identify any potential imbalances across the different product categories. By examining the proportion of each sentiment type, we can assess whether there is a predominance of one sentiment over another and evaluate if the dataset is skewed towards a particular sentiment. This information is crucial for ensuring that our analysis is balanced and representative of the underlying data.



The bar chart above displays the top 20 positive words used in product reviews, with "good" being the most frequently mentioned, followed by "like" and "nice," indicating common positive sentiments expressed by customers. This highlights the prevalent terms that contribute to positive product feedback.

The bar chart below shows the top 20 negative words found in product reviews, with issue and problem being the most frequently mentioned, followed by noise, indicating common negative sentiments and problems experienced by customers. This highlights the prevalent terms that contribute to negative product feedback.



In our statistical analysis, we observed a slight positive correlation between overall ratings and both the rating count and the weighted rating. This indicates that products with higher ratings tend to attract more reviews and achieve higher weighted ratings. Additionally, we found a moderate positive correlation (0.12) between the "rating" and "discounted_price" variables, suggesting that products available at lower discounted prices often receive higher customer ratings.

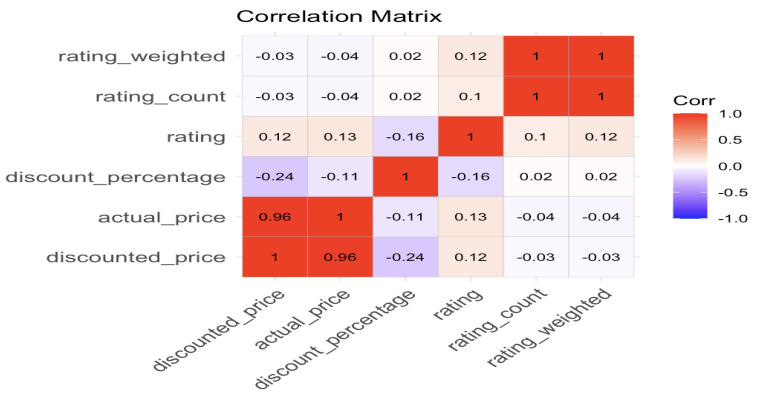


Fig: Plot of correlation matrix of numerical columns

7. SENTIMENT ANALYSIS

Final goal is to perform sentiment analysis to gain insights of ratings, reviews over products, categories and build a predicting model. After EDA, we need to pre-process the dataset for sentiment analysis. This requires pre-processing of data by cleaning and transforming the text data. Data pre-processing involves following stages:

Text Cleaning: We conducted a series of text cleaning procedures to eliminate noise, irrelevant characters, and special symbols from the text data. This involved removing punctuation, digits, and special characters, as well as converting all text to lowercase.

Stopword Removal: We filtered out common stopwords from the text data. These stopwords are frequently occurring words (such as "the," "and" and "is") that lack substantial meaning and can be omitted from the analysis without affecting the overall results.

Lemmatization or Stemming: To standardise the text, we applied lemmatization or stemming techniques to reduce words to their base or root forms. This process helps in decreasing the dimensionality of the dataset and enhancing the performance of the sentiment analysis model.

Feature Engineering: We developed additional features and generated new variables from the text data, including metrics such as word count, average word length, and sentiment scores. These sentiment scores were derived using external libraries such as **syuzhet.**

Sentiment labelling is the next phase, where we categorise each review in the dataset according to its sentiment. Typically, this involves classifying reviews into one of three sentiment categories: positive, negative, or neutral, based on the overall sentiment expressed in the text.

The primary tasks involved in sentiment labelling include:

1. Assigning a sentiment label (positive, negative, or neutral) to each review based on its content. This process can be performed manually by human annotators or automatically using machine learning models trained on labelled data.
2. After sentiment labelling is complete, each review will have a corresponding sentiment label. This allows us to move on to text vectorization and the construction of sentiment models.

cleaned_reviews <chr>	sentiment_score <dbl>	sentiment <chr>
look durable charge fine toono complainscharging really fast good producttill now satisfy qualitythis good product charge speed slow original iphone cablegood quality recommendhttpsmmediaamaz...	3.850000e+00	positive
order cable connect phone android auto car cable really strong connection port really good make already micro usb cable ambrane still good shape connect phone car use cable get connect good iss...	6.100000e+00	positive
quite durable sturdyhttpsmmediaamazoncomimageswwebptimagesiriggrbuclsyjpgworking goodhttpsmmediaamazoncomimageswwebptimagesibkpyowlsyjpgproductvery nice productworking wellits ...	1.250000e+00	positive
good productlong wirecharges goodnicei buy cable r worthy product price test various charger adapter w w support fast charge wellgoodoki get good price sale amazon product useful warranty warr...	4.100000e+00	positive
buy instead original apple work r fast apple charger good option want cheap good product buy ipad pro work flawlessly build quality okay like gonna hang clothe want strong cable even braid cable s...	4.050000e+00	positive
good productlikevery good item strong useful usb cablevalue moneythanks amazon producerhttpsmmediaamazoncomimagesizrelsyjpggoodnice product useful productsturdy support w charge	2.250000e+00	positive
build quality good come year warrantygood productbought charge mobile tab doesnt work lenovo be tabguys cable good compare everyone heat protection quickly charge chance shock circuitgoodn...	1.400000e+00	positive
worth money suitable android auto purpose serve cargot rseverything okay package good feel like seller give use cablegood productgood product cost moreoriginal cablei buy cable use cable androi...	6.300000e+00	positive
use connect old pc internet try lubuntu ubuntu work box didnt setup theres extender cable can place comfortable place get model antenna otherwise youll range problem youre directly line sight wifi...	1.120000e+01	positive
order cable connect phone android auto car cable really strong connection port really good make already micro usb cable ambrane still good shape connect phone car use cable get connect good iss...	6.100000e+00	positive

Fig: Cleaned reviews and respective sentiment score labels.

Text Vectorization: Upon cleaning reviews and labelling we need to vectorize as Machine learning algorithms cannot directly process raw text data, so we need to transform the language into numerical representations. Common text vectorization techniques include Bag-of-Words, TF-IDF (Term Frequency-Inverse Document Frequency), and word embeddings (such as Word2Vec and GloVe). These methods convert each review into a vector or matrix of numerical features that can be used by the models.

Continued to build model with following stages:

Feature Extraction: During text vectorization, we extract significant features from the text data to capture the reviews' semantics and context. These characteristics may be Word frequencies, n-grams (near word sequences), and dense vector representations derived from word embeddings are some examples. The features selected are determined by the intricacy of the text data and the sentiment analysis model's performance requirements.

Model Selection: Once the text input has been vectorized, we choose suitable machine learning techniques or deep learning architectures for sentiment analysis. Logistic regression, support vector machines (SVM), random forests, recurrent neural networks (RNNs), and convolutional neural networks (CNNs) are among the most widely used models. Model selection is determined by criteria such as dataset size, text feature complexity, and desired prediction accuracy.

7.1 Model Training

After selecting the model, we proceed to train it using the vectorized text data along with the corresponding sentiment labels. During this training phase, the model learns to recognize patterns in text features that correspond to various sentiment categories (positive, negative, and neutral). The training algorithm fine-tunes the model parameters to minimise prediction errors and enhance the accuracy of sentiment predictions. For model training have considered splitting the data into 80:20 ratio for training and testing respectively.

```
set.seed(123)
trainIndex <- createDataPartition(cleaned_df$sentiment, p = .8, list = FALSE, times = 1)
trainData <- df_tfidf[trainIndex,]
testData <- df_tfidf[-trainIndex,]
trainLabels <- cleaned_df$sentiment[trainIndex]
testLabels <- cleaned_df$sentiment[-trainIndex]
```

7.1.1 Support Vector Machine (SVM)

Support Vector Machines (SVM) are powerful supervised learning algorithms used for both classification and regression tasks. Their goal is to identify an optimal hyperplane that maximally separates different classes within the feature space. SVMs are effective in high-dimensional settings and can manage nonlinear separations through the use of various kernel functions, such as linear, polynomial, and radial basis function (RBF) kernels.

Key Features of SVM

1. Hyperplane Construction: SVM identifies a hyperplane in a high-dimensional space that separates different classes with the largest possible margin. The hyperplane's defining elements are the support vectors, which are the closest data points to the hyperplane and play a crucial role in establishing the decision boundaries.
2. Classification and Regression: For classification tasks, SVM aims to maximise the distance between classes, which reduces the likelihood of classification errors. In regression tasks, known as Support Vector Regression (SVR), the algorithm seeks to fit as many data points as possible within a specified margin of error, ensuring the model generalises well to new data.

The hyperparameter to configure is the cost value, which dictates the margin around the separating hyperplanes.

```
Overall Statistics

               Accuracy : 0.9784
                 95% CI : (0.9536, 0.992)
    No Information Rate : 0.9676
    P-Value [Acc > NIR] : 0.2023

                  Kappa : 0.4918

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: negative Class: neutral Class: positive
Sensitivity                  0.33333             NA          1.0000
Specificity                  1.00000              1          0.3333
Pos Pred Value               1.00000             NA          0.9782
Neg Pred Value               0.97818             NA          1.0000
Prevalence                   0.03237              0          0.9676
Detection Rate               0.01079              0          0.9676
Detection Prevalence         0.01079              0          0.9892
Balanced Accuracy            0.66667             NA          0.6667
```

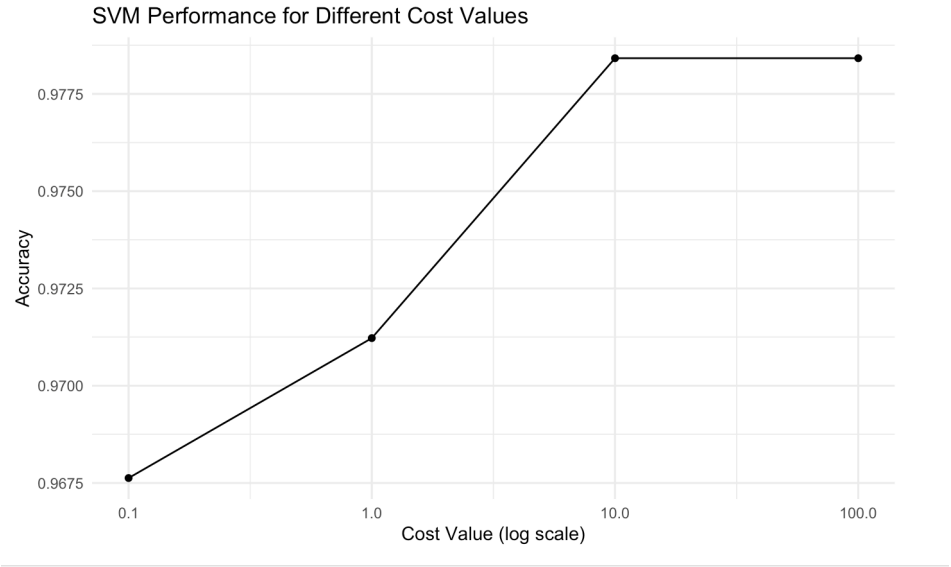
Fig: SVM Confusion Matrix

	Cost <dbl>	Accuracy <dbl>
Accuracy	0.1	0.9676259
Accuracy1	1.0	0.9712230
Accuracy2	10.0	0.9784173
Accuracy3	100.0	0.9784173

Fig: SVM performance for different cost values

The SVM model's accuracy varies with different cost values, which adjust the margin around the separating hyperplanes. The results indicate that as the cost parameter C increases from 0.1

to 10.0, the accuracy of the SVM model improves, going from 96.76% to 97.84%. However, further increasing C from 10.0 to 100.0 does not improve the accuracy, which stays at 97.84%.



7.1.2 Random Forest

Random Forest is a robust ensemble machine learning algorithm that enhances prediction accuracy and stability by combining multiple decision trees. Utilising the bagging technique, it constructs each tree using random subsets of the data and features, which reduces correlation among trees and mitigates overfitting. The process begins with generating bootstrap samples from the original dataset, with each tree being trained on a different sample. Additionally, at each decision node, a random subset of features is considered, further diversifying the trees. For classification, predictions are made based on the majority vote of the trees, while for regression, the average or median of the predictions is used. Although Random Forest excels in handling large, high-dimensional datasets and provides valuable insights into feature importance, it can be computationally demanding and lacks interpretability due to the complexity of combining numerous decision trees.

Key Features of Random Forest

- **Ensemble Method**: Random Forest is based on the concept of combining multiple models to improve overall performance. Each decision tree in the forest votes for a class, and the majority vote determines the final classification.
- **Bootstrapping**: The algorithm generates multiple datasets by randomly sampling with replacement from the original dataset. Each decision tree is trained on a different bootstrap sample.
- **Feature Randomness**: During the construction of each tree, a random subset of features is selected for splitting nodes. This introduces diversity among the trees and helps in reducing overfitting.

For the random forest, it's essential to determine the number of features to use for each split in the decision tree. A common guideline is to use the square root of the total number of features. After experimenting, decided on using 4, 8, 12, 16, and 20 features.

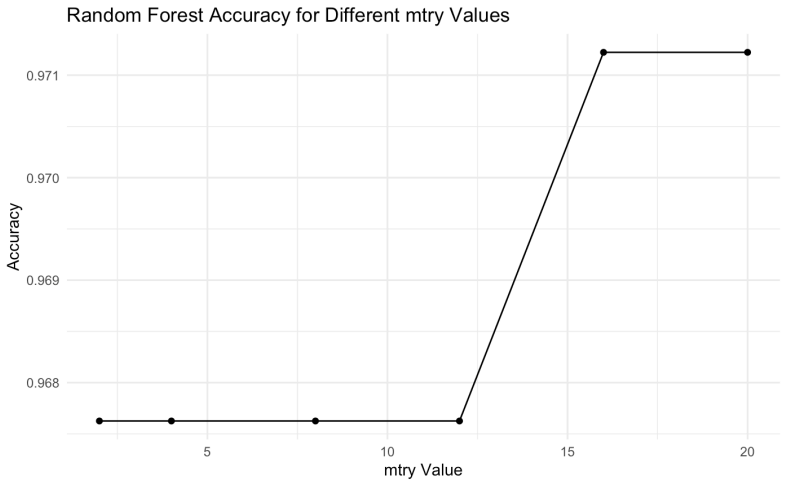
```
Overall Statistics

               Accuracy : 0.9712
                 95% CI : (0.9441, 0.9875)
    No Information Rate : 0.9676
    P-Value [Acc > NIR] : 0.4535

                  Kappa : 0.1948

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: negative Class: neutral Class: positive
Sensitivity                 0.111111             NA          1.0000
Specificity                 1.000000              1          0.1111
Pos Pred Value              1.000000             NA          0.9711
Neg Pred Value              0.971119             NA          1.0000
Prevalence                  0.032374              0          0.9676
Detection Rate              0.003597              0          0.9676
Detection Prevalence        0.003597              0          0.9964
Balanced Accuracy           0.555556             NA          0.5556
```

Fig: Random Forest Confusion Matrix

	mtry <dbl>	Accuracy <dbl>
Accuracy	2	0.9676259
Accuracy1	4	0.9676259
Accuracy2	8	0.9676259
Accuracy3	12	0.9676259
Accuracy4	16	0.9712230
Accuracy5	20	0.9712230

Fig: Random Forest performance for different features

The results of the random forest model, evaluated with different numbers of features (mtry) for each split, show consistent accuracy levels. For mtry values of 2, 4, 8, and 12, the model achieved an accuracy of 96.76%. Increasing mtry to 16 and 20 slightly improved the accuracy to 97.12%. This indicates that while the choice of mtry has a modest impact on model performance, using a higher mtry value can lead to a small increase in accuracy.



7.1.3 K-Nearest Neighbors:

K-Nearest Neighbors (KNN) is a straightforward, yet powerful algorithm commonly used in statistical pattern recognition and machine learning for classification and regression tasks. At its heart, KNN predicts the classification of a data point based on the most frequent class among its closest neighbors. The 'K' in KNN denotes the number of nearest neighbors considered for making the prediction.

The KNN algorithm relies on similarity or distance metrics, such as Euclidean, Manhattan, or Minkowski distance. When classifying a new data point, the algorithm calculates the distances between this point and all others in the dataset. It then identifies the K nearest points and assigns the new data point the class that appears most frequently among these neighbors.

A significant benefit of KNN is its simplicity and effectiveness in handling multi-class problems. Since it doesn't make any assumptions about the underlying data distribution, KNN is a non-parametric method. This feature is particularly advantageous in real-world situations where the data might not fit typical theoretical distributions.

The sole hyperparameter for this model is k, which represents the number of neighbors taken into account.

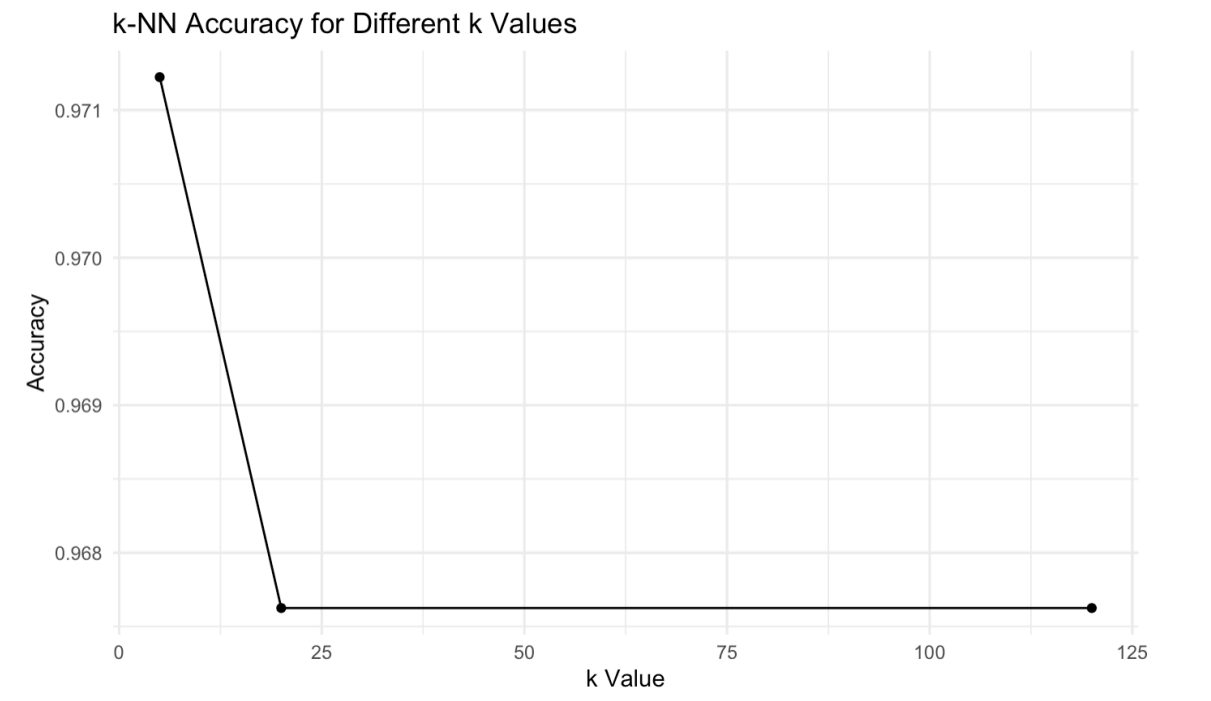
	k <dbl>	Accuracy <dbl>
Accuracy	5	0.9712230
Accuracy1	20	0.9676259
Accuracy2	120	0.9676259

Fig: KNN performance for different K values

```
Overall Statistics

               Accuracy : 0.9676
                 95% CI : (0.9394, 0.9851)
    No Information Rate : 0.9676
    P-Value [Acc > NIR] : 0.5874

                  Kappa : 0

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: negative Class: neutral Class: positive
Sensitivity                  0.00000             NA          1.0000
Specificity                  1.00000              1          0.0000
Pos Pred Value                   NaN             NA          0.9676
Neg Pred Value               0.96763             NA             NaN
Prevalence                   0.03237              0          0.9676
Detection Rate               0.00000              0          0.9676
Detection Prevalence         0.00000              0          1.0000
Balanced Accuracy            0.50000             NA          0.5000
```

Fig: KNN Confusion Matrix

The k-nearest neighbors (k-NN) model was evaluated using different values of k to determine its impact on accuracy. When k was set to 5, the model achieved the highest accuracy of 97.12%. However, increasing k to 20 and 120 resulted in a slight decrease in accuracy to 96.76%. These results suggest that a smaller k value (5) is optimal for this dataset, providing the best balance between bias and variance, while larger k values may lead to a slight drop in model performance.



Comparison of models:

The performance of the three models, SVM, Random Forest, and k-NN, was evaluated using a dataset with three classes: negative, neutral, and positive. The overall accuracy, confidence intervals, and class-specific statistics were compared to determine the strengths and weaknesses of each model.

The SVM model demonstrated high overall accuracy, particularly excelling in correctly identifying positive classes with perfect sensitivity and negative predictive value. However, it showed a lower sensitivity for the negative class.

The Random Forest model had a slightly lower overall accuracy compared to the SVM. While it achieved perfect specificity and positive predictive value for the negative class, it struggled with sensitivity for the same class. This indicates the model's tendency to correctly identify positives but miss some negative instances.

The k-NN model exhibited the lowest overall accuracy among the three models. It showed perfect specificity but failed to identify any negative instances (sensitivity = 0.0000), resulting in a kappa value of 0, indicating no agreement between the predicted and actual classes beyond chance.

MACHINE LEARNING OPTIMIZATION

Machine learning optimization is essential for improving the performance of our sentiment analysis models. It focuses on fine-tuning various parameters to maximise accuracy and efficiency. We start by choosing the most appropriate algorithms for sentiment analysis and perform below optimization.

Hyperparameter Tuning: This involves adjusting parameters like learning rates and regularisation strengths to balance bias and variance, thereby enhancing model performance.

Cross-Validation Strategies: To ensure our models generalise well and avoid overfitting, we use cross-validation. This technique involves dividing the dataset into subsets, training and validating the models iteratively, which helps in assessing their robustness and reliability.

8. PREDICTIVE MODELLING

Predicting discount percentages and rating is a critical task in retail and marketing analytics. Accurate forecasting of discounts can help businesses optimise pricing strategies, maximize revenue, and improve customer satisfaction. We developed and evaluated a Random Forest Regression model to predict discount percentages based on product attributes such as actual price and category.

Predicting rating:

```r
# Select relevant columns and create a subset for modeling
pred_model_data <- cleaned_df %>%
  select(discounted_price, actual_price, rating_count, rating)

# Ensure numerical columns are treated as numeric
pred_model_data$discounted_price <- as.numeric(as.character(pred_model_data$discounted_price))
pred_model_data$actual_price <- as.numeric(as.character(pred_model_data$actual_price))
pred_model_data$rating <- as.numeric(as.character(pred_model_data$rating))
pred_model_data$rating_count <- as.numeric(as.character(pred_model_data$rating_count))
```

```r
# Split data into training and test sets
set.seed(123)
trainIndex <- createDataPartition(pred_model_data$rating, p = 0.7, list = FALSE)
trainData <- pred_model_data[trainIndex, ]
testData <- pred_model_data[-trainIndex, ]

```

```r
# Train Random Forest model
rf_model <- randomForest(rating ~ ., data = trainData, importance = TRUE)
```

```r
# Predict on test set
rf_predictions <- predict(rf_model, newdata = testData)
```

```r
# Evaluate model performance
rf_rmse <- RMSE(rf_predictions, testData$rating)
print(paste("RMSE (Random Forest):", rf_rmse))
```

```
[1] "RMSE (Random Forest): 0.263041181021724"
```

An RMSE of 0.26 suggests that the model's predictions are relatively accurate. A lower RMSE generally reflects better model performance. Thus, the model provides valuable insights for optimising rating predictions and can be integrated into business decision-making processes.

Predicting discount percentage:

```r
# Select relevant columns and create a subset for modeling
discount_data <- cleaned_df %>%
  select(discount_percentage, actual_price, category)

# Ensure numerical columns are treated as numeric
discount_data$actual_price <- as.numeric(as.character(discount_data$actual_price))
discount_data$discount_percentage <- as.numeric(as.character(discount_data$discount_percentage))

# Convert 'category' to factor if categorical
discount_data$category <- as.factor(discount_data$category)

# Split data into training and test sets
set.seed(123)
trainIndex <- createDataPartition(discount_data$discount_percentage, p = 0.7, list = FALSE)
trainData <- discount_data[trainIndex, ]
testData <- discount_data[-trainIndex, ]

# Train Random Forest model
rf_discount_model <- train(discount_percentage ~ ., data = trainData, method = "rf")

# Predict on test set
discount_predictions <- predict(rf_discount_model, newdata = testData)

# Evaluate model performance
rmse_discount <- RMSE(discount_predictions, testData$discount_percentage)
print(paste("RMSE for Discount Percentage:", rmse_discount))
```

```
[1] "RMSE for Discount Percentage: 0.159091659698273"
```

An RMSE of 0.159 suggests that the model's predictions are relatively accurate. A lower RMSE generally reflects better model performance. Thus, the model provides valuable insights for optimising discount strategies and can be integrated into business decision-making processes.

9. RECOMMENDATION SYSTEM

Recommendation System serves as a valuable tool for companies to gain deeper customer insights, elevate user engagement and satisfaction, and drive revenue growth through strategic product recommendations.

User-Based Collaborative Filtering is a recommendation approach that predicts a user's interest in items based on the preferences of similar users. This method operates under the assumption that users who agreed in the past will agree in the future about item preferences. The core mechanism involves measuring the similarity between users' ratings and leveraging this information to predict ratings for items not yet rated by a user.

We have developed a User-Based Collaborative Filtering (UBCF) recommendation system leveraging machine learning algorithms to suggest products tailored to users' interests and prior purchases based on their past review and ratings.

```r
```{r}
get_product_details <- function(product_ids, product_info_df) {
  product_info_df %>%
    filter(product_id %in% product_ids) %>%
    select(product_id, product_name, category)
}

# Converting the dataframe to a matrix format suitable for recommenderlab
rating_matrix1 <- dcast(cleaned_df_long, user_id_con ~ product_id, value.var = "rating", fun.aggregate = mean)
rownames(rating_matrix1) <- rating_matrix1$user_id_con
rating_matrix1$user_id_con <- NULL

# Converting to realRatingMatrix
rating_matrix1 <- as(as.matrix(rating_matrix1), "realRatingMatrix")

set.seed(123)
split <- sample(x = c(TRUE, FALSE), size = nrow(rating_matrix1), replace = TRUE, prob = c(0.8, 0.2))
trainData <- rating_matrix1[split, ]
testData <- rating_matrix1[!split, ]

# Training a user-based collaborative filtering model
ubcf_model <- Recommender(trainData, method = "UBCF")

# Making predictions for a specific user
id_search <- "AE55KTFVNXYFD5FPYWP2OUPEYNPQ"
```

```r
  # Finding the index of the user in the trainingData based on the user_id column
  user_index <- which(rownames(trainData) == id_search)
if (length(user_index) > 0) {
    # Checking if the user has rated any items
    user_ratings <- trainData[user_index, ]
    if (length(user_ratings@data) > 0) {
      cat("User Ratings:\n")

      # Extracting the rated items and their ratings
      rated_items <- as(user_ratings, "list")[[1]]
      for (item in names(rated_items)) {
        cat("Product ID:", item, "Rating:", rated_items[[item]], "\n")
      }
      # Making predictions for the specified user
      predictions <- predict(ubcf_model, user_ratings, n = 100) # Recommend top 100 products
     # Checking if there are recommendations
      if (length(predictions@items) > 0) {
        # Viewing recommendations
        recommended_items <- as(predictions, "list")
        recommended_items_info <- get_product_details(recommended_items, cleaned_df_long)
        cat("Recommendations:\n")
        for (i in 1:length(recommended_items)) {
          # cat("Product ID:", recommended_items[[i]], "\n")
          cat("Product ID:", recommended_items[[i]], "\n",
              "Name:", recommended_items_info$product_name[i], "\n",
              "Category:", recommended_items_info$category[i], "\n")
        }
      } else {
        print("No recommendations found.")
      }
    } else {
      print("The user has not rated any items.")
    }
} else {
    print("User ID not found in the training data.")
}
```
```

The model showcases the implementation of a user-based collaborative filtering (UBCF) recommendation system. It begins by preparing the data, including converting a user-product rating dataset into a format suitable for the recommender lab package. The dataset is then split into training and testing sets to evaluate model performance. Using the training data, a UBCF model is trained to predict user preferences based on similar users' ratings. For a specific user, the model identifies their rated items and generates personalized recommendations. If the user has provided ratings, the code retrieves and displays the recommended products along with their details. This process demonstrates how collaborative filtering can be used to make tailored product suggestions based on user interactions and preferences.

Recommendations for invalid user:

```
# Making predictions for a specific user
id_search <- "AE55KTFVNXYFD5FPYWP2OUPEYNPQa"
```

```
[1] "User ID not found in the training data."
```

Recommendations for valid user based on past purchase history:

```
# Making predictions for a specific user
id_search <- "AE55KTFVNXYFD5FPYWP2OUPEYNPQ"
```

```
User Ratings:
Product ID: B07N8RQ6W7 Rating: 4.1
Product ID: B08N1WL9XW Rating: 4
Product ID: B08P9RYPLR Rating: 4
Product ID: B09NHVCHS9 Rating: 4
Product ID: B09NJN8L25 Rating: 4
Product ID: B09NKZXMWJ Rating: 4
Product ID: B09NL4DJ2Z Rating: 4
Product ID: B0B3MQXNFB Rating: 4
Product ID: B0B3N8VG24 Rating: 4
Recommendations:
Product ID: B087FXHB6J
 Name: Zebronics Zeb-Companion 107 USB Wireless Keyboard and Mouse Set with Nano Receiver (Black)
 Category: Computers&Accessories|Accessories&Peripherals|Keyboards,Mice&InputDevices|Keyboard&MouseSets
```

10. CONCLUSION

By analyzing the purchasing patterns of individual users, we can uncover their preferences for specific product categories and features. This information allows us to refine product design and craft targeted marketing strategies to appeal to customer segments. Additionally, the system enhances customer engagement by recommending products that align with users' interests, thereby enriching their shopping experience and fostering greater satisfaction. This, in turn, contributes to improved customer retention and loyalty. Furthermore, the system’s

ability to suggest similar or complementary products to those already purchased boosts opportunities for cross-selling and up-selling, ultimately leading to increased revenue and profitability for the company.

In future work, exploring advanced machine learning techniques and ensemble methods could significantly enhance the accuracy and robustness of sentiment analysis models. For the recommendation system, integrating hybrid approaches that combine collaborative filtering with content-based methods may offer more personalized and accurate recommendations. We wanted to implement deep learning techniques to improve the overall performance, but due to inconsistencies and limitations with R, we could not implement efficiently.

Source Code:

https://github.com/satwikas/DPA_Project

REFERENCES

1. Sultana, Najma & Kumar, Pintu & Patra, Monika & Chandra, Sourabh & Alam, Sk. (2019). SENTIMENT ANALYSIS FOR PRODUCT REVIEW. International Journal of Soft Computing. 09. 7. 10.21917/ijsc.2019.0266.

2. N. Ul Saaqib, Gunika and H. K. Verma, "Analysis of Sentiment on Amazon Product Reviews," 2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC), Jalandhar, India, 2023, pp. 697-702, doi: 10.1109/ICSCCC58608.2023.10176787.

3. Haque, Tanjim & Saber, Nudrat & Shah, Faisal. (2018). Sentiment analysis on large scale Amazon product reviews.10.1109/ICIRD.2018.8376299.

4. Dublin, Griffith & Joseph, Roshan. (2020). Amazon Reviews Sentiment Analysis: A Reinforcement Learning Approach. 10.13140/RG.2.2.31842.35523.