MAHATMA EDUCATION SOCIETY’S

PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE

(Autonomous)

NEW PANVEL

PROJECT REPORT ON

**“Analyzing Supermarket Sales Trends for Profitable Insights”**

IN PARTIAL FULFILLMENT OF

BACHELOR OF INFORMATION TECHNOLOGY

SEMESTER VI– 2023-24

PROJECT GUIDE

Prof. Omkar Sherkhane

SUBMITTED BY: Athulkrishna Pramod

ROLL NO: 3105

Mahatma Education Society's  
**PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE (Autonomous)  
Re-accredited “A” Grade by NAAC (3rd Cycle)**

A logo of a college

Description automatically generated

**Project Completion Certificate**

**THIS IS TO CERTIFY THAT**

**Mr. Athulkrishna Pramod**

of **M.Sc. Data Analytics Part – I** has completed the project titled **“Analyzing Supermarket Sales Trends for Profitable Insights”** of subject **Big Data Analytics** under our guidance and supervision during the academic year 2023-24 in the department of Computer Science.

Head of the Department

Project Guide

Course Coordinator

**Introduction**

The growth of supermarkets in most populated cities is increasing and market competitions are also high. Within the retail sector, supermarkets stand at the forefront of leveraging Big Data analytics to understand consumer behavior and enhance business strategies. This project aims to explore the application of advanced analytical techniques to a dataset sourced from a supermarket chain.

Through this project, we hope to demonstrate the transformative impact of Big Data analytics in enhancing the operational efficiency, profitability, and customer satisfaction levels of supermarket chains. By leveraging data-driven insights, businesses can gain a competitive edge in a highly competitive marketplace, ultimately driving growth and success in the ever-evolving retail landscape.

Ultimately, this project aims to demonstrate the transformative impact of data-driven insights in enhancing operational efficiency, profitability, and customer satisfaction levels within the supermarket industry.

**Tools and Techniques**

For the implementation of this project, we employed a combination of tools and techniques to conduct exploratory data analysis (EDA) and develop machine learning (ML) models. Google Colab, a cloud-based Jupyter notebook environment provided by Google, served as our primary platform for executing Python code, facilitating seamless collaboration and computation with its integrated support for popular libraries such as Pandas, NumPy, Matplotlib, and Scikit-learn.

**Tools –**

* **Google Colab:**

Google Colab offers a range of benefits for data analysis and machine learning tasks. Its integration with Google Drive enables easy access to datasets and notebooks, ensuring data security and availability. Moreover, Colab provides access to powerful GPUs and TPUs, allowing for accelerated computation, particularly beneficial for training complex machine learning models.

* **Dataset:**

**supermarket\_sales.csv -** The dataset is from Kaggle.

It is one of the historical sales of supermarket companies which has recorded in 3 different branches for 3 months data. Predictive data analytics methods are easy to apply with this dataset.

This dataset typically includes information such as the date and time of the transaction, the products that were purchased, the price of each product, the total amount spent on the transaction, and other relevant details.

**Dataset Link –**

<https://www.kaggle.com/datasets/aungpyaeap/supermarket-sales>

**Techniques –**

* **Exploratory Data Analysis (EDA):**

For EDA, we utilized the Pandas library for data manipulation and preprocessing, along with Matplotlib and Seaborn for data visualization. These tools enabled us to gain insights into the structure, distribution, and relationships within the supermarket sales dataset. Through descriptive statistics, histograms, box plots, and correlation matrices, we identified patterns, outliers, and potential areas for further analysis.

* **Machine Learning Models:**

In the realm of machine learning, we employed Scikit-learn, a versatile library for building and evaluating ML models. Leveraging algorithms such as linear regression, decision trees, and random forests, we developed predictive models to forecast future sales trends based on historical data. Additionally, we utilized techniques such as feature engineering and cross-validation to optimize model performance and mitigate overfitting.

By using Google Colab along with the tools mentioned earlier, we made the process of analyzing data and creating models much easier. This helped us to quickly understand the data and build models that could predict future sales. These insights we gained can now be used to make smarter decisions for improving supermarket sales strategies.

**Summary of Columns in the Dataset**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Types | Description |
| Invoice ID | object | Computer generated sales slip invoice identification number |
| Branch | object | Branch of supercenter (3 branches are available identified by A, B and C) |
| City | object | Location of supercenters |
| Customer type | object | Type of customers, recorded by Members for customers using member card and Normal for without member card |
| Gender | object | Gender type of customer |
| Product line | object | General item categorization groups - Electronic accessories, Fashion accessories, Food and beverages, Health and beauty, Home and lifestyle, Sports and travel |
| Unit price | float64 | Price of each product in $ |
| Quantity | Int64 | Number of products purchased by customer |
| Tax 5% | float64 | 5% tax fee for customer buying |
| Total | float64 | Total price including tax |
| Date | object | Date of purchase (Record available from January 2019 to March 2019) |
| Time | object | Purchase time (10am to 9pm) |
| Payment | object | Payment used by customer for purchase (3 methods are available – Cash, Credit card and E-wallet) |
| cogs | float64 | Cost of goods sold |
| gross margin percentage | float64 | margin percentage: Gross margin percentage |
| gross income | float64 | Gross income |
| Rating | float64 | Customer stratification rating on their overall shopping experience (On a scale of 1 to 10) |

**Code & Output**

**Importing Important Libraries –**

from google.colab import drive

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import matplotlib.colors as mcolors

%matplotlib inline

drive.mount('/content/drive')

**Import Dataset –**

file = '/content/drive/My Drive/BD\_Project/supermarket\_sales.csv'

df = pd.read\_csv(file)

**Exploring the Dataset –**

df.head()

A screenshot of a phone

Description automatically generated

df.tail()

A screenshot of a computer

Description automatically generated

df.shape



df.columns

A black text on a white background

Description automatically generated

df.info()

A screenshot of a computer screen

Description automatically generated

df.describe()

A table of numbers and a tax

Description automatically generated with medium confidence

**Data Cleaning –**

df.isnull().sum()

A screenshot of a computer screen

Description automatically generated

df.duplicated().sum()



There is Nothing to clean, let’s move on to the next step that is Exploratory Data Analysis (EDA).

**Exploratory Data Analysis (EDA) –**

In this step I would like to explore these questions to analyze the trends in Supermarket sales.

1. What is the average total amount spent by each customer type and gender?
2. What are the total sales for each gender and product line combination?
3. What are the most popular payment methods used in the supermarket?
4. Which cities are the biggest contributors to the overall sales?
5. What are the most profitable product lines in the supermarket?
6. What is the average total amount spent by each customer type and gender?

For Customer Type –

# group the data by customer type and calculate the average total amount spent by each type

df\_customer\_type = df.groupby('Customer type')['Total'].mean()

print(df\_customer\_type)

# Importing the required libraries

import matplotlib.pyplot as plt

# Adjusting the size of the plot

plt.figure(figsize=(8, 6))

# Plotting the bar chart

df\_customer\_type.plot(kind='bar', color=['skyblue', 'lightgreen'], width=0.6)

# Adding title and labels

plt.title('Average Total Amount Spent By Customer Type')

plt.xlabel('Customer Type')

plt.ylabel('Average Total Amount Spent')

# Rotating the x-axis labels for better readability

plt.xticks(rotation=0)

# Displaying the plot

plt.tight\_layout()

plt.show()

**Output –**

A number of numbers and letters

Description automatically generated with medium confidence

A graph of a number of people

Description automatically generated

The analysis indicates that there is a slight difference in the average total amount spent between customers with membership cards (Member) and those without (Normal). On average, members spend approximately $9.67 more than non-members per transaction. This finding suggests that membership status may influence customers' spending behavior, with members tending to spend slightly more.

For Gender Type –

# group the data by gender and calculate the average total amount spent by each gender

df\_genderr = df.groupby('Gender')['Total'].mean()

print(df\_genderr)

# Importing the required libraries

import seaborn as sns

import matplotlib.pyplot as plt

# Adjusting the size of the plot

plt.figure(figsize=(8, 6))

# Plotting the box plot

sns.boxplot(x='Gender', y='Total', data=df, palette={'Female': 'lightblue', 'Male': 'lightcoral'})

# Adding title and labels

plt.title('Distribution of Total Amount Spent By Gender')

plt.xlabel('Gender')

plt.ylabel('Total Amount Spent')

# Displaying the plot

plt.tight\_layout()

plt.show()

**Output –**

A number on a white background

Description automatically generated

A diagram of a distribution of amount spent by gender

Description automatically generated

This graph says that on an average, female customers appear to spend notably more per transaction compared to their male counterparts, with an approximate difference of $25. This suggests potential gender-specific preferences or buying behaviors within the dataset.

1. What are the total sales for each gender and product line combination?

# group the data by gender and product line and calculate the total sales for each group

df\_gender\_product\_line = df.groupby(['Gender', 'Product line'])['Total'].sum()

print(df\_gender\_product\_line)

# Importing the required libraries

import matplotlib.pyplot as plt

# Adjusting the size of the plot

plt.figure(figsize=(12, 8))

# Plotting the bar chart

df\_gender\_product\_line.plot(kind='bar', colormap='viridis', alpha=0.75)

# Adding title and labels

plt.title('Total Sales for Each Group of Gender and Product Line')

plt.xlabel('Gender and Product Line Group')

plt.ylabel('Total Sales')

# Rotating the x-axis labels for better readability

plt.xticks(rotation=45, ha='right')

# Displaying the plot

plt.tight\_layout()

plt.show()

**Output –**

A screenshot of a computer

Description automatically generated

A graph of purple vertical lines

Description automatically generated with medium confidence

# Importing the required libraries

import seaborn as sns

import matplotlib.pyplot as plt

plt.figure(figsize=(11, 6))

# Plotting the bar plot

sns.barplot(x='Product line', y='Total', hue='Gender', data=df, estimator=sum, ci=None, palette='pastel')

# Adding title and labels

plt.title('Total Sales for Each Product Line Grouped by Gender')

plt.xlabel('Product Line')

plt.ylabel('Total Sales')

# Rotating the x-axis labels for better readability

plt.xticks(rotation=45, ha='right')

# Displaying the plot

plt.tight\_layout()

plt.show()

A graph of sales

Description automatically generated

The analysis reveals distinct purchasing patterns between genders across various product lines. Female customers demonstrate higher sales in Fashion accessories, Food and beverages, and Home and lifestyle categories, while male customers contribute more to Health and beauty products. These insights can guide tailored marketing strategies and product offerings to better meet the preferences of different gender segments.

1. What are the most popular payment methods used in the supermarket?

# group the data by payment method and calculate the total sales for each payment method

payment\_method\_sales = df.groupby('Payment')['Total'].sum()

# sort the payment methods by total sales in descending order

payment\_method\_sales = payment\_method\_sales.sort\_values(ascending=False)

# print the most popular payment methods

print('Most popular payment methods:')

print(payment\_method\_sales.head(10))

# Importing the required libraries

import matplotlib.pyplot as plt

# Adjusting the size of the plot

plt.figure(figsize=(8, 4))

# Plotting the horizontal bar chart

payment\_method\_sales.head(10).plot(kind='barh', color='skyblue')

# Adding title and labels

plt.title('Most Popular Payment Methods')

plt.xlabel('Total Sales')

plt.ylabel('Payment Method')

# Displaying the plot

plt.tight\_layout()

plt.show()

**Output –**

A close-up of a card

Description automatically generated

A bar graph with blue and white stripes

Description automatically generated

The analysis of payment methods demonstrates that Cash emerges as the preferred choice among consumers, with E-wallet and Credit card following closely behind. Recognizing these trends is crucial for businesses to tailor their strategies effectively. By offering a diverse range of payment options, companies can better cater to consumer preferences, thereby enhancing satisfaction levels and optimizing sales performance.

1. Which cities are the biggest contributors to the overall sales?

# Importing the required libraries

import pandas as pd

# Calculating the total sales for each city

df\_city\_sales = df.groupby('City')['Total'].sum()

# Printing the total sales for each city

print("Total Sales by City:")

print(df\_city\_sales)

# Importing the required library

import matplotlib.pyplot as plt

# Adjusting the size of the plot

plt.figure(figsize=(10, 6))

# Plotting the bar chart with a different color (e.g., green)

df\_city\_sales.plot(kind='bar', color='lightgreen')

# Adding title and labels

plt.title('Total Sales by City')

plt.xlabel('City')

plt.ylabel('Total Sales')

# Rotating the x-axis labels for better readability

plt.xticks(rotation=0)

# Displaying the plot

plt.tight\_layout()

plt.show()

**Output –**

A screenshot of a computer

Description automatically generated

A green bar chart with white text

Description automatically generated

In this analysis, we examined total sales figures across different cities. Results indicate that Chicago leads with $110,568.71 in sales, closely followed by San Diego with $106,200.37 and Austin with $106,197.67. Understanding regional sales variations is essential for devising targeted marketing strategies and resource allocation, ensuring sustained business growth and market competitiveness.

1. What are the most profitable product lines in the supermarket?

# group the data by product line and calculate the total sales for each product line

product\_line\_sales = df.groupby('Product line')['Total'].sum()

# sort the product lines by total sales in descending order

product\_line\_sales = product\_line\_sales.sort\_values(ascending=False)

# print the most popular product lines

print('Most popular product lines:')

print(product\_line\_sales.head(10))

# Importing the required library

import matplotlib.pyplot as plt

# Defining a light color palette

colors = ['#FF9999', '#66B2FF', '#99FF99', '#FFCC99', '#FFD700', '#87CEEB', '#FFB6C1', '#98FB98', '#FF6347', '#C0C0C0']

# Adjusting the size of the plot

plt.figure(figsize=(8, 8))

# Plotting the pie chart with the light color palette

plt.pie(product\_line\_sales.head(10), labels=product\_line\_sales.head(10).index, autopct='%3.1f%%', startangle=140, colors=colors)

# Adding title

plt.title('Most Popular Product Lines')

# Displaying the plot

plt.axis('equal')

plt.tight\_layout()

plt.show()

**Output –**

A screenshot of a computer

Description automatically generated

A pie chart with text on it

Description automatically generated

The analysis of product lines uncovered Food and beverages as the leading category in sales, closely trailed by Sports and travel, electronic accessories, Fashion accessories, Home and lifestyle, and Health and beauty. These insights offer valuable guidance for inventory management and marketing strategies, enabling businesses to cater effectively to consumer preferences and capitalize on lucrative sales opportunities.

**Correlation matrix –**

import seaborn as sns

import matplotlib.pyplot as plt

# Calculate correlation matrix

correlation\_matrix = df.corr()

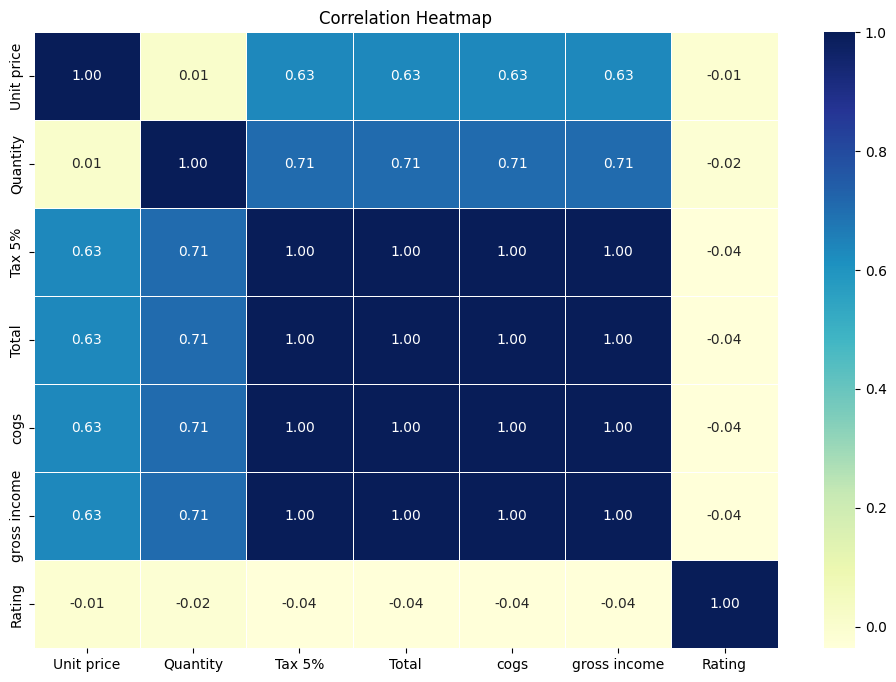
plt.figure(figsize=(12, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='YlGnBu', fmt=".2f", linewidths=.5)

plt.title("Correlation Heatmap")

plt.show()

**Output –**



The correlation analysis reveals several noteworthy relationships among the variables in our dataset. Particularly, we observe a strong positive correlation between variables such as Unit price, Tax 5%, Total, and cogs, indicating that higher unit prices correspond to increased tax, total sales, and cost of goods sold. However, no significant correlation is observed with the gross margin percentage. Additionally, the correlation analysis underscores the importance of considering these interdependencies when interpreting the dataset and making informed business decisions.

**Data Preparation –**

Removing Unwanted features

# Dropping unnecessary columns

df.drop(['Invoice ID'], axis=1, inplace=True)

**Model Building –**

In the model building, three prominent regression models—Linear Regression, Decision Tree Regression, and Random Forest Regression (RFR)—have been employed to predict total sales.

In the end, these models will be compared based on their performance metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R2), allowing for the selection of the best-fit model to predict total sales effectively from the dataset.

**Linear Regression –**

# Importing required libraries

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

# Splitting the dataset into features (X) and target variable (y)

X = df.drop('Total', axis=1)

y = df['Total']

# One-hot encode categorical variables

X\_encoded = pd.get\_dummies(X, drop\_first=True)

# Splitting the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_encoded, y, test\_size=0.2, random\_state=42)

# Initializing and fitting the linear regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Predicting on the testing set

y\_pred = model.predict(X\_test)

# Evaluating the model

mse = mean\_squared\_error(y\_test, y\_pred)

mae = mean\_absolute\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error (MSE):", mse)

print("Mean Absolute Error (MAE):", mae)

print("R-squared (R2):", r2)

**Output –**



**Graph for this model –**

# Calculating residuals

residuals = y\_test - y\_pred

# Creating residual plot

plt.figure(figsize=(8, 6))

plt.scatter(y\_pred, residuals, color='skyblue', alpha=0.7)

# Adding a horizontal line at y=0

plt.axhline(y=0, color='red', linestyle='--')

# Adding labels and title

plt.title('Residual Plot (Linear Regression)')

plt.xlabel('Predicted Total Sales')

plt.ylabel('Residuals')

# Displaying the plot

plt.tight\_layout()

plt.show()

A graph with blue dots

Description automatically generated

The linear regression model demonstrates exceptional performance, with a negligible mean squared error (MSE) of 4.83e-12 and mean absolute error (MAE) of 4.75e-07. Furthermore, the model achieves a near-perfect R-squared value of 1.0, indicating that it explains virtually all the variance in the target variable. These results suggest that the model accurately predicts total sales based on the provided features, offering valuable insights for business decision-making and forecasting.

**Decision Tree –**

import pandas as pd

from sklearn.tree import DecisionTreeRegressor

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error, r2\_score

X = df.drop('Total', axis=1)

y = df['Total']

X\_encoded = pd.get\_dummies(X, drop\_first=True)

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_encoded, y, test\_size=0.2, random\_state=42)

dct\_model = DecisionTreeRegressor(random\_state=42)

dct\_model.fit(X\_train, y\_train)

# Make predictions

y\_predDT = dct\_model.predict(X\_test)

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_predDT)

mae = mean\_absolute\_error(y\_test, y\_predDT)

r2 = r2\_score(y\_test, y\_predDT)

print("Mean Squared Error (MSE):", mse)

print("Mean Absolute Error (MAE):", mae)

print("R-squared (R2):", r2)

**Output –**

A group of black text

Description automatically generated

**Graph for this model –**

# Calculating residuals

residuals = y\_test - y\_predDT

# Creating residual plot

plt.figure(figsize=(8, 6))

plt.scatter(y\_predDT, residuals, color='skyblue', alpha=0.7)

# Adding a horizontal line at y=0

plt.axhline(y=0, color='red', linestyle='--')

# Adding labels and title

plt.title('Residual Plot (Decision Tree)')

plt.xlabel('Predicted Total Sales')

plt.ylabel('Residuals')

# Displaying the plot

plt.tight\_layout()

plt.show()

A graph of blue dots

Description automatically generated

The Decision Tree regression model demonstrates strong predictive performance, with a mean squared error (MSE) of 4.53 and mean absolute error (MAE) of 1.30. Moreover, it achieves an impressive R-squared value of approximately 0.9999, indicating its robust ability to explain the variance in total sales. These results underscore the effectiveness of the Decision Tree approach in capturing intricate patterns within the dataset, enhancing its utility for predictive analytics and business insights.

**Random Forest Regression –**

from sklearn.ensemble import RandomForestRegressor

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error

from sklearn.metrics import r2\_score

X = df.drop('Total', axis=1)

y = df['Total']

X\_encoded = pd.get\_dummies(X, drop\_first=True)

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_encoded, y, test\_size=0.2, random\_state=42)

ran\_model = RandomForestRegressor(n\_estimators=100, random\_state=42)

ran\_model.fit(X\_train, y\_train)

# Make predictions

y\_predRF = ran\_model.predict(X\_test)

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_predRF)

mae = mean\_absolute\_error(y\_test, y\_predRF)

r2 = r2\_score(y\_test, y\_predRF)

print("Mean Squared Error (MSE):", mse)

print("Mean Absolute Error (MAE):", mae)

print("R-squared (R2):", r2)

**Output –**

A group of numbers and symbols

Description automatically generated

**Graph for this model –**

import matplotlib.pyplot as plt

# Calculate residuals

residuals = y\_test - y\_predRF

# Create predicted vs. residual plot

plt.figure(figsize=(8, 6))

plt.scatter(y\_predRF, residuals, color='skyblue', alpha=0.7)

# Add horizontal line at y=0

plt.axhline(y=0, color='red', linestyle='--')

# Add labels and title

plt.title('Residual Plot (Random Forest Regression)')

plt.xlabel('Predicted Total Sales')

plt.ylabel('Residuals')

# Display the plot

plt.tight\_layout()

plt.show()

A graph with blue dots

Description automatically generated

The Random Forest regression model exhibits outstanding performance in predicting total sales. With a remarkably low mean squared error (MSE) of 1.59 and mean absolute error (MAE) of 0.74, it demonstrates exceptional accuracy. Moreover, its high R-squared value of approximately 0.9999 indicates that it effectively captures the variance in total sales, making it a reliable and robust model for forecasting.

**Conclusion**

Among the three models evaluated for predicting total sales, **Random Forest Regression emerged as the most robust and accurate**. While Linear Regression achieved impressive results with minimal errors and a perfect fit to the data, Random Forest Regression surpassed it, demonstrating significantly lower errors and a superior predictive performance, as evidenced by its remarkably low Mean Squared Error (MSE), Mean Absolute Error (MAE), and high R-squared value. Decision Tree Regression, although commendable, exhibited a slightly higher error rate compared to Linear Regression and Random Forest Regression. Therefore, in practical applications where precision and reliability are paramount, Random Forest Regression stands out as the preferred choice, offering businesses a powerful tool to optimize resource allocation and revenue generation strategies.

And if we look at the EDA part the analysis of various aspects related to sales behavior and consumer preferences reveals several key insights. Membership status appears to influence spending behavior, with members spending slightly more on average compared to non-members. Gender-specific spending patterns indicate that female customers tend to spend more per transaction, particularly in categories like Fashion accessories and Food and beverages, while males show a preference for Health and beauty products.

Additionally, payment method preferences suggest that offering diverse payment options, with a focus on Cash, E-wallet, and Credit card, can enhance consumer satisfaction and drive sales. Furthermore, understanding regional sales variations and product line performance is crucial for devising targeted marketing strategies and optimizing inventory management to capitalize on sales opportunities and ensure business growth.