

UNIFIED MENTOR



Supermarket Grocery Sales - Retail Analytics Report

Submitted by – Ishita

Intern ID - UMIP277158

Internship Domain- Data Science Intern

Introduction

Every day, the supermarket industry produces vast amounts of sales data that reveal information about consumer buying patterns, product demand, and regional performance. Businesses can improve their operations by gaining important insights from the analysis of this data about inventory management, sales patterns, and customer preferences.

Supermarkets deal with issues such seasonal sales fluctuations, inventory control, and shifting consumer demand. Businesses may make well-informed decisions that increase profitability, boost customer satisfaction, and optimize supply chain operations by utilizing data analytics. The goal of this project is to methodically handle and examine supermarket sales data in order to find important insights that can inform strategic business choices. We will investigate a number of supermarket sales-related topics through this analysis, including determining the top-selling items, comprehending regional variations in sales performance, and forecasting future sales patterns. The findings will also be made easier to understand through the use of visualization approaches. Supermarket managers and business analysts will benefit from the study's conclusions as they make data-driven decisions, enhance their marketing plans, and guarantee the best possible product availability.

The complete data analysis process will be covered in this paper, including exploratory analysis, machine learning modeling, performance evaluation, and data preprocessing. The objective is to show how data analytics can turn unstructured sales data into insightful knowledge that helps the retail industry.

Data Preprocessing

1. Handling Missing Values:

- Missing values were identified using functions like `is.na()` and `sum(is.na())`.
- Rows with missing values were removed to ensure the integrity of the dataset. This was essential as missing data can lead to biased or inaccurate results.

2. Removing Duplicates:

- Duplicate entries were checked using `duplicated()` and removed to prevent uneven results. This ensures that each observation in the dataset is unique and contributes equally to the analysis.

3. Encoding Categorical Variables:

- Categorical variables were converted into numerical formats using techniques such as one-hot encoding or label encoding. This allows the model to interpret categorical data effectively.

4. Date Formatting:

- Date values were converted to appropriate date formats using the `as.Date()` function. Proper date formatting is crucial for conducting time-series analysis, as it allows for accurate chronological sorting and manipulation of date-related data.

Exploratory Data Analysis (EDA)

1. Sales Distribution Across Categories:

- Histograms and bar plots were created using ggplot2 to visualize the distribution of sales across different categories. This helps identify which categories perform best and which may need further analysis or improvement.

```
ggplot(data, aes(x = Category, y = Sales)) +  
geom_bar(stat = 'identity', fill = 'blue') +  
ggtitle('Sales Distribution by Category') +  
xlab('Category') +  
ylab('Total Sales')
```

2. Trends Over Time:

- Time-series plots were generated to examine sales trends over time. Line graphs were used to illustrate how sales fluctuate month by month or year by year, providing insights into seasonal trends and overall growth.

```
ggplot(sales_over_time, aes(x = Order.Date, y = Total.Sales)) +  
geom_line(color = 'green') +  
ggtitle('Sales Trends Over Time') +  
xlab('Order Date') +  
ylab('Total Sales')
```

3. Correlation Between Numeric Features:

- Correlation matrices and scatter plots were utilized to explore relationships between numeric features. This analysis helps identify potential predictors for the sales model.

```
cor_matrix <- cor(numeric_data)

ggplot(data = as.data.frame(cor_matrix), aes(x = Var1, y = Var2)) +
  geom_tile(aes(fill = value), color = 'white') +
  scale_fill_gradient2(low = 'blue', high = 'red', mid = 'white', limit = c(-1, 1), name =
'Correlation') +
  theme_minimal() +
  ggtitle('Correlation Matrix')
```

Predictive Modeling

1. Splitting the Dataset:

- The dataset was divided into training and testing sets using the `createDataPartition()` function. An 80-20 split was utilized, ensuring that the model is trained on a substantial portion of the data while retaining a separate set for validation.

```
set.seed(42)

train_index <- createDataPartition(target, p = 0.8, list = FALSE)

X_train <- features[train_index, ]
X_test <- features[-train_index, ]

y_train <- target[train_index]
y_test <- target[-train_index]
```

2. Normalization:

- The training data was normalized using centering and scaling methods. Normalization ensures that all features contribute equally to the model training, preventing features with larger scales from dominating the learning process.

```
preprocess_params <- preProcess(X_train, method = c("center", "scale"))

X_train <- predict(preprocess_params, X_train)
```

```
X_test <- predict(preprocess_params, X_test)
```

3. Training the Model:

- A linear regression model was trained using the `train()` function from the `caret` package. The model learned the relationship between the features and the target variable (sales).

```
model <- train(X_train, y_train, method = 'lm')
```

4. Making Predictions:

- Predictions were generated for the test set using the trained model. This step is crucial for evaluating the model's performance.

```
y_pred <- predict(model, X_test)
```

5. Model Evaluation:

- The model's performance was evaluated using Mean Squared Error (MSE) and R-squared metrics. MSE measures the average squared difference between predicted and actual values, while R-squared indicates the proportion of variance explained by the model.

```
mse <- mean((y_test - y_pred)^2, na.rm = TRUE)
```

```
r2 <- cor(y_test, y_pred, use = "complete.obs")^2
```

```
print(paste('Mean Squared Error:', mse))
```

```
print(paste('R-squared:', r2))
```

R – CODE

```
# Load Libraries
```

```
library(ggplot2)
```

```
library(dplyr)
```

```
library(lubridate)
```

```
library(caret)
```

```
library(corrplot)
```

```
library(readr)
```

```
library(reshape2)
```

```
# Load Dataset
```

```
df <- read.csv("C:/Users/ishit/Downloads/supermarket dataset unified/Supermart Grocery  
Sales - Retail Analytics Dataset.csv", stringsAsFactors = TRUE)
```

```
# Data Preprocessing
```

```
df <- na.omit(df)
```

```
df <- df[!duplicated(df), ]
```

```
# Convert Order Date
```

```
df$Order.Date <- as.Date(df$Order.Date, format='%Y-%m-%d')
```

```
df$Order.Day <- day(df$Order.Date)
```

```
df$Order.Month <- month(df$Order.Date)
```

```
df$Order.Year <- year(df$Order.Date)
```

```
# Encode categorical variables
```

```
df$Category <- as.numeric(factor(df$Category))
```

```
df$Sub.Category <- as.numeric(factor(df$Sub.Category))
```

```
df$City <- as.numeric(factor(df$City))
```

```
df$Region <- as.numeric(factor(df$Region))
```

```
df$State <- as.numeric(factor(df$State))
```

```
df$Sales <- as.numeric(df$Sales)
```

```
# Exploratory Data Analysis
```

```
# Boxplot
```

```
ggplot(df, aes(x=factor(Category), y=Sales)) +
```

```
  geom_boxplot(fill='lightblue') +
```

```
  ggtitle('Sales Distribution by Category') +
```

```
  xlab('Category') +
```

```
  ylab('Sales')
```

```
# Ensure Order.Date is in Date format
```

```
df$Order.Date <- as.Date(df$Order.Date)
```

```
# Filter out NA values and summarize profit
```

```
profit_over_time <- df %>%
```

```
  filter(!is.na(Profit)) %>% # Remove NA values
```

```
  group_by(Order.Date) %>%
```

```
  summarise(Total.Profit = sum(Profit, na.rm = TRUE))
```

```
# Plot the profit data as a bar plot
```

```
ggplot(profit_over_time, aes(x = Order.Date, y = Total.Profit)) +
```

```
  geom_bar(stat = 'identity', fill = 'green') + # Use bar plot with filled color
```

```
  ggtitle('Total Profit Over Time') +
```

```
  ylab('Total Profit') +
```

```
  xlab('Order Date') +
```

```
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) # Rotate x-axis labels for better visibility
```



```
# Correlation Matrix
```

```
cor_matrix <- cor(df %>% select_if(is.numeric))
```

```
corrplot(cor_matrix, method='color', type='upper', tl.cex=0.8)
```

```
# Sales by City
```

```
ggplot(df, aes(x=factor(City), y=Sales)) +
```

```
  geom_bar(stat='summary', fun=mean, fill='skyblue') +
```

```
  ggtitle('Average Sales by City') +
```

```
  xlab('City') +
```

```
  ylab('Average Sales')
```

```
# Sales by Sub-Category
```

```
ggplot(df, aes(x=factor(Sub.Category), y=Sales)) +
```

```
  geom_boxplot(fill='lightgreen') +
```

```
  ggtitle('Sales Distribution by Sub-Category') +
```

```
  xlab('Sub-Category') +
```

```
  ylab('Sales')
```

```
# Pie Chart: Region-wise Sales
```

```
region_sales <- df %>% group_by(Region) %>% summarise(Total = sum(Sales))
```

```
region_sales$Region <- as.factor(region_sales$Region)
```

```
pie(region_sales$Total, labels=region_sales$Region, main="Sales Distribution by Region")
```

```
# Feature Selection and Model Building
```

```
features <- df %>% select(-c(Order.ID, Customer.Name, Order.Date, Sales))
```

```
target <- df$Sales
```

```
target
```

```
#split the dataset
```

```
set.seed(42)

train_index <- createDataPartition(target, p=0.8, list=FALSE)
X_train <- features[train_index, ]
X_test <- features[-train_index, ]
y_train <- target[train_index]
y_test <- target[-train_index]

# Normalize Data
preprocess_params <- preProcess(X_train, method=c("center", "scale"))
X_train <- predict(preprocess_params, X_train)
X_test <- predict(preprocess_params, X_test)

# Train Linear Regression Model
model <- train(X_train, y_train, method='lm')
y_pred <- predict(model, X_test)

# Check for NA values in predictions and actuals
if (any(is.na(y_test)) || any(is.na(y_pred))) {
  stop("NA values found in predictions or actuals.")
}

# Evaluate the Model
mse <- mean((y_test - y_pred)^2, na.rm = TRUE) # Ensure NA removal
r2 <- cor(y_test, y_pred, use = "complete.obs")^2 # Use complete cases for correlation

# Print results
print(paste('Mean Squared Error:', mse))
print(paste('R-squared:', r2))
```

Actual vs Predicted Plot

```
ggplot(data.frame(Actual=y_test, Predicted=y_pred), aes(x=Actual, y=Predicted)) +
```

```
geom_point(color='blue') +
```

```
geom_abline(slope=1, intercept=0, color='red') +
```

```
ggtitle('Actual vs Predicted Sales') +
```

```
xlab('Actual Sales') +
```

```
ylab('Predicted Sales')
```

Outputs

```
80
81 # Feature Selection and Model Building
82 features <- df %>% select(-c(Order.ID, Customer.Name, Order.Date, Sales))
83 target <- df$Sales
84 target
85
86 # Split the dataset
87 set.seed(42)
88 train_index <- createDataPartition(target, p=0.8, list=FALSE)
89 X_train <- features[train_index, ]
90 X_test <- features[-train_index, ]
91 y_train <- target[train_index]
92 y_test <- target[-train_index]
```

847 (Top Level)

Console Terminal Background Jobs

R433 ~/#

```
> plot(region_sales_train, lwd=2, main="Region Sales by Region")
> features <- df %>% select(-c(Order.ID, Customer.Name, Order.Date, Sales))
> target <- df$Sales
> target
[1] 1254 749 2360 896 2355 2305 826 1847 791 1795 1903 701 1659 1277 831 1440 1678 1617
[19] 1757 692 522 948 707 969 1100 2022 541 979 1988 989 1845 1400 2163 1689 809 1265
[37] 934 1354 1751 2045 660 1868 1017 2311 1663 1420 1415 1488 726 1259 902 2306 1417 627
[55] 760 2350 1179 1624 1316 2121 1428 706 921 1741 607 1824 1368 1160 1294 1232 994 2468
[73] 724 2375 1965 1489 1250 1358 719 2303 2027 1739 990 2354 1044 2129 1739 1663 2114 1878
[91] 769 1241 2128 2257 1285 1230 1856 545 2301 1475 2118 1527 567 969 2300 1575 540 536
[109] 556 1179 1487 2220 2155 1154 2420 1305 927 983 903 1235 1243 649 2354 1443 975 889
[127] 2352 1293 1458 1801 1261 2112 640 1658 2373 1430 1873 867 1101 1088 866 2483 1335 765
[145] 2403 1551 1564 1125 768 1985 2287 1016 852 1233 2236 2188 1114 1774 2364 1487 713 737
[163] 1004 1516 1442 1690 2021 2340 705 1215 552 1412 791 2476 1747 1973 2025 1016 689 1322
[181] 1605 1718 2177 578 782 1388 2264 813 2015 1433 2093 2149 1972 2060 1175 1006 1409 914
[199] 1352 682 1738 2421 2245 896 1033 1998 1469 1899 2194 1942 609 2353 1591 1353 1020 1362
[217] 1733 1949 2368 2110 1940 1557 1443 2024 2168 621 1230 560 1286 1242 717 1480 826 1846
[235] 2045 1983 650 2163 1481 2492 962 1432 523 1540 817 2462 962 538 1556 2133 1474 531
[253] 1769 1257 1899 1929 2417 1443 660 2114 1278 867 943 1836 1200 1167 696 2500 1507 1612
[271] 764 884 2443 1147 1899 744 812 862 1061 2365 1421 1357 1007 1544 1537 1530 2460 976
[289] 1106 1165 1244 1092 1396 849 2488 840 1820 1969 2261 1793 644 1359 1534 2090 931 1157
[307] 2396 1677 590 1401 678 2417 2211 2392 1835 2281 1583 1123 1079 1027 1771 1171 672 848
[325] 2078 1400 945 590 1583 1534 2133 2021 1825 2029 1228 1501 1094 1179 1127 601 1697 2095
[343] 1132 1561 1238 2068 1113 1966 2138 978 2014 645 2194 1244 743 2474 1494 1174 1062 2297
[361] 562 2049 570 1236 1069 1946 595 2439 2415 1725 688 1698 1665 1309 905 2271 2091 1917
[379] 641 1192 1132 1727 1051 2109 1311 759 2201 1964 1898 2282 2294 2213 1797 1912 1865 1838
[397] 1626 647 1042 1519 532 1477 707 1748 2257 762 904 958 2193 830 1531 2355 917 1433
[415] 2246 2130 2461 1863 1382 807 1272 1621 1356 684 540 1256 581 707 913 507 1367 833
[433] 1987 1152 1362 942 2200 861 1648 1805 620 1216 686 1648 506 1579 550 2020 695 2453
[451] 1820 508 2459 850 2417 795 2142 1542 2411 1213 507 618 1199 2200 1238 1962 580 2412
[469] 1224 1356 502 2271 1977 1388 2115 1976 1323 639 1544 1223 1224 2157 673 2092 1892 1943
[487] 1968 1596 1637 2232 1917 531 1283 937 1373 1598 1398 1070 1948 702 1953 548 1952 1836
[505] 2292 2112 2354 2061 1791 2009 1850 898 1070 606 2352 1381 1170 1815 595 640 1854 2088
[523] 1837 737 2447 1102 2236 1413 1032 1597 1537 1245 1902 888 1416 739 1262 2241 2060 1633
[541] 1338 1382 1296 2289 898 2017 1404 888 659 1167 1471 2481 2336 1696 1904 861 1505 1791
[559] 1826 768 1551 1313 1455 777 2179 1527 2207 1879 1668 727 979 1362 1626 681 2017 980
[577] 931 1454 2139 1456 2403 822 750 1483 1072 842 1479 619 1784 1586 1354 1545 2409 1432
[595] 1443 1770 1298 1674 1925 1963 1899 2172 2289 2110 1577 2092 2269 1426 1895 580 791 1167
[613] 1138 885 1149 753 2493 2236 2106 1873 1155 1881 2285 561 1482 977 1595 1069 566 1551
[631] 1279 745 540 620 1931 2151 1547 926 1419 534 2037 966 1125 2333 1361 1656 1351 992
[649] 559 2404 1082 1051 1578 1086 1703 1630 1689 1972 1797 1562 2398 2372 1527 884 2482 1250
[667] 1548 2147 2159 1054 883 1676 2323 1292 2421 1204 2118 1832 1136 1952 1686 2202 2387 1709
[685] 2340 1130 960 2142 774 2427 2231 1163 520 713 2128 2302 1735 2178 2173 1064 1259 2086
[703] 2062 2463 506 1354 1030 530 1409 1419 2371 1078 1084 1498 1797 1596 2019 2146 544 1271
[721] 762 1362 617 2130 1272 661 2290 1152 2128 2042 1819 2310 1405 2103 2155 1399 1332 2014
[739] 1808 2033 698 759 700 550 1454 1194 837 516 2238 2032 1842 1589 1531 1641 1586 1655
[757] 1958 1083 2031 1617 1758 1881 1658 1401 1949 2239 1903 1367 1000 535 1562 1114 2382 649
[775] 1880 825 1026 1273 1072 1794 1889 2196 1856 2406 1746 704 1787 825 1702 1167 1059 1546
[793] 2135 1802 979 816 663 1627 1549 2018 1983 2029 1980 2375 1736 1843 1176 662 1501 1900
```

```

111
112 # Print results
113 print(paste('Mean Squared Error:', mse))
114 print(paste('R-squared:', r2))
115
116
117 # Actual vs Predicted Plot
118 ggplot(data.frame(Actual=y_test, Predicted=y_pred), aes(x=Actual, y=Predicted)) +
119   geom_point(color='blue') +
120   geom_abline(slope=1, intercept=0, color='red') +
121   ggtitle('Actual vs Predicted Sales') +
122   xlab('Actual Sales') +
123   ylab('Predicted Sales')

```

114:31 (Top Level) ✖ Copilot: Not signed in

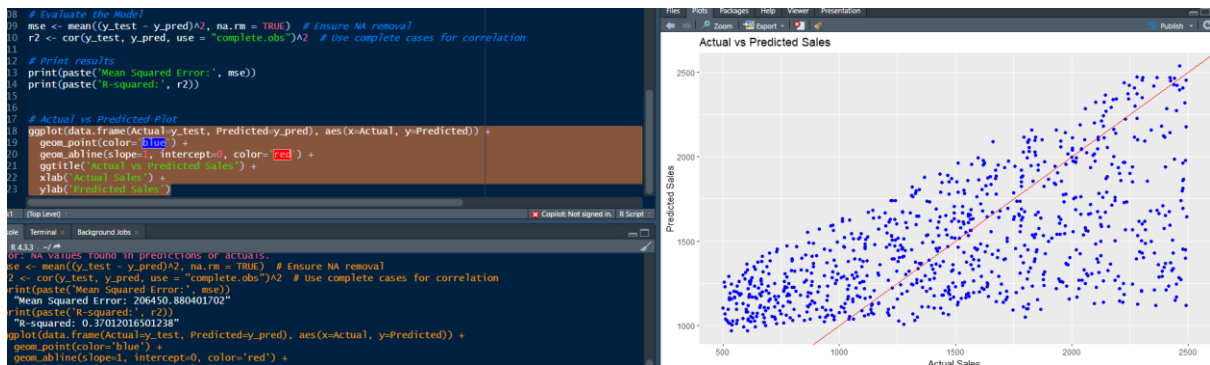
Console Terminal Background Jobs

R 4.3.3 ~ /

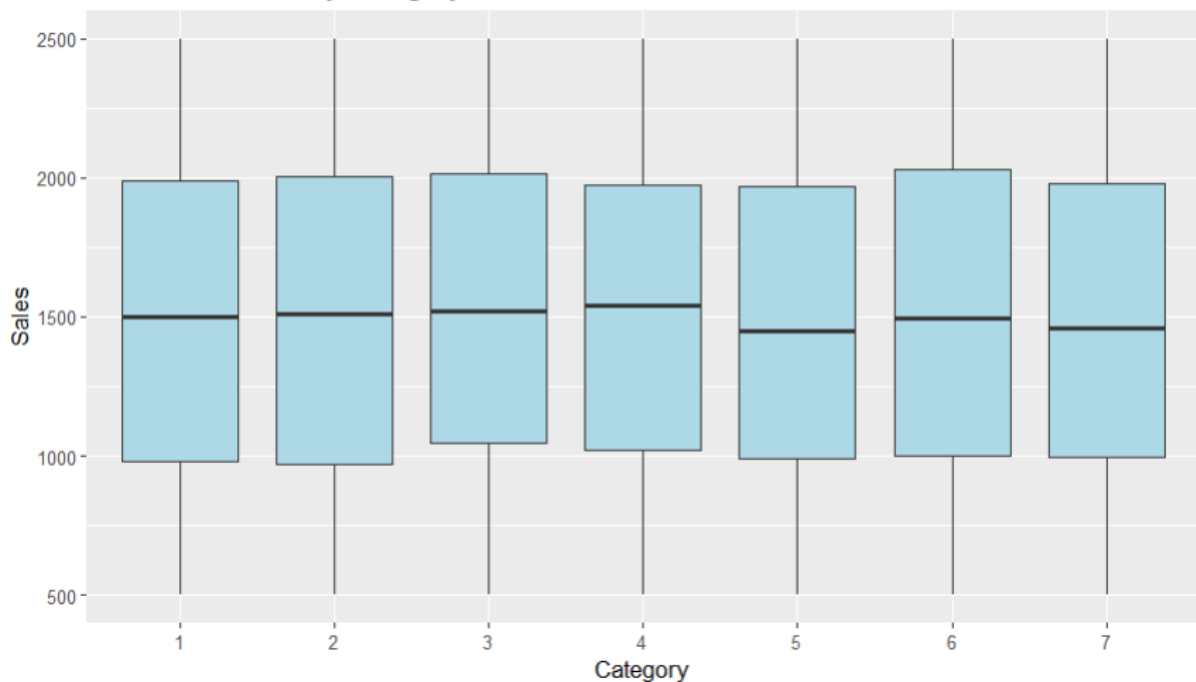
```

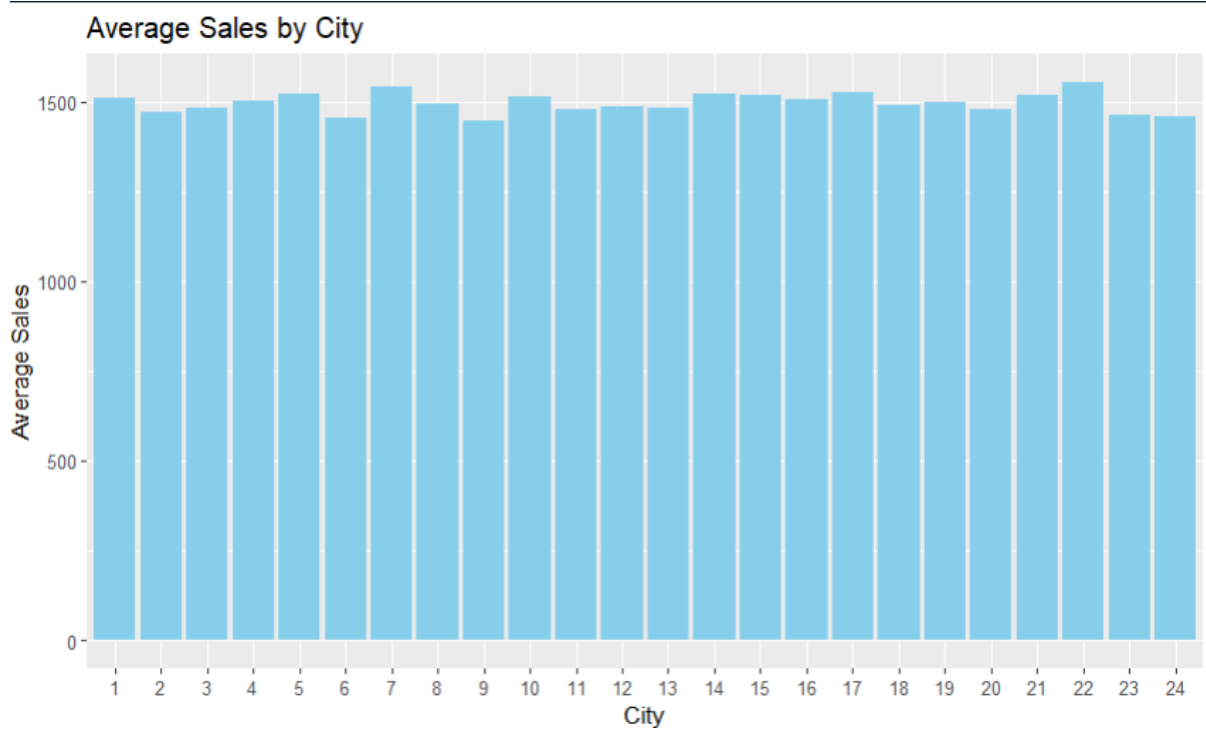
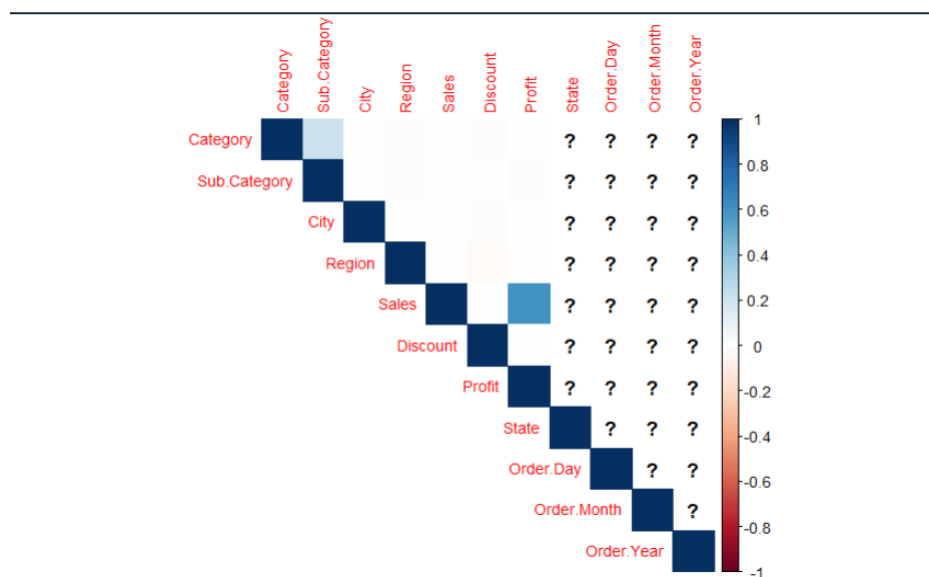
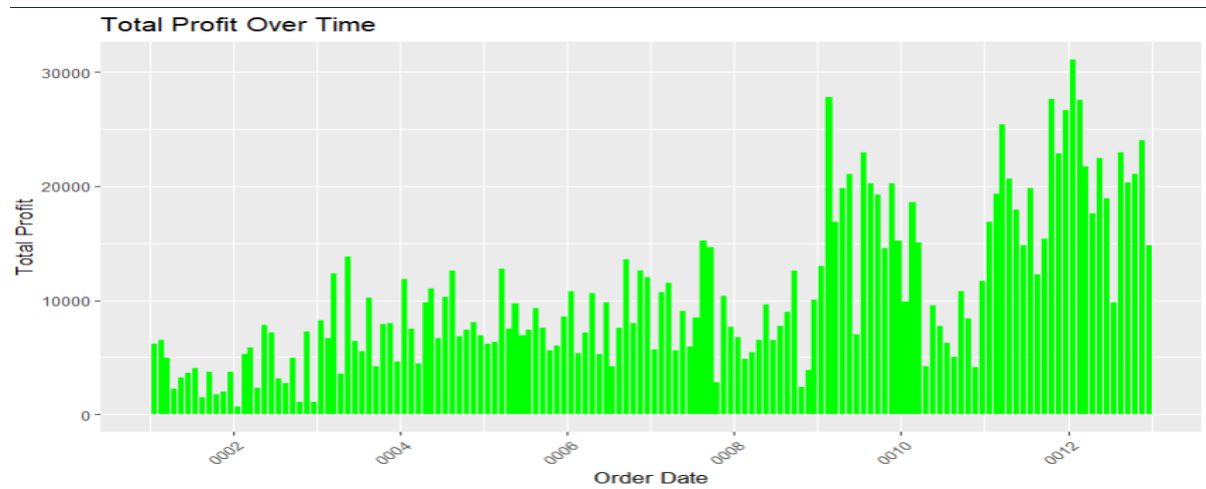
> r2 <- cor(y_test, y_pred, use = "complete.obs")^2 # Use complete cases for correlation
> print(paste('Mean Squared Error:', mse))
[1] "Mean Squared Error: 206450.880401702"
> print(paste('R-squared:', r2))
[1] "R-squared: 0.37012016501238"
>

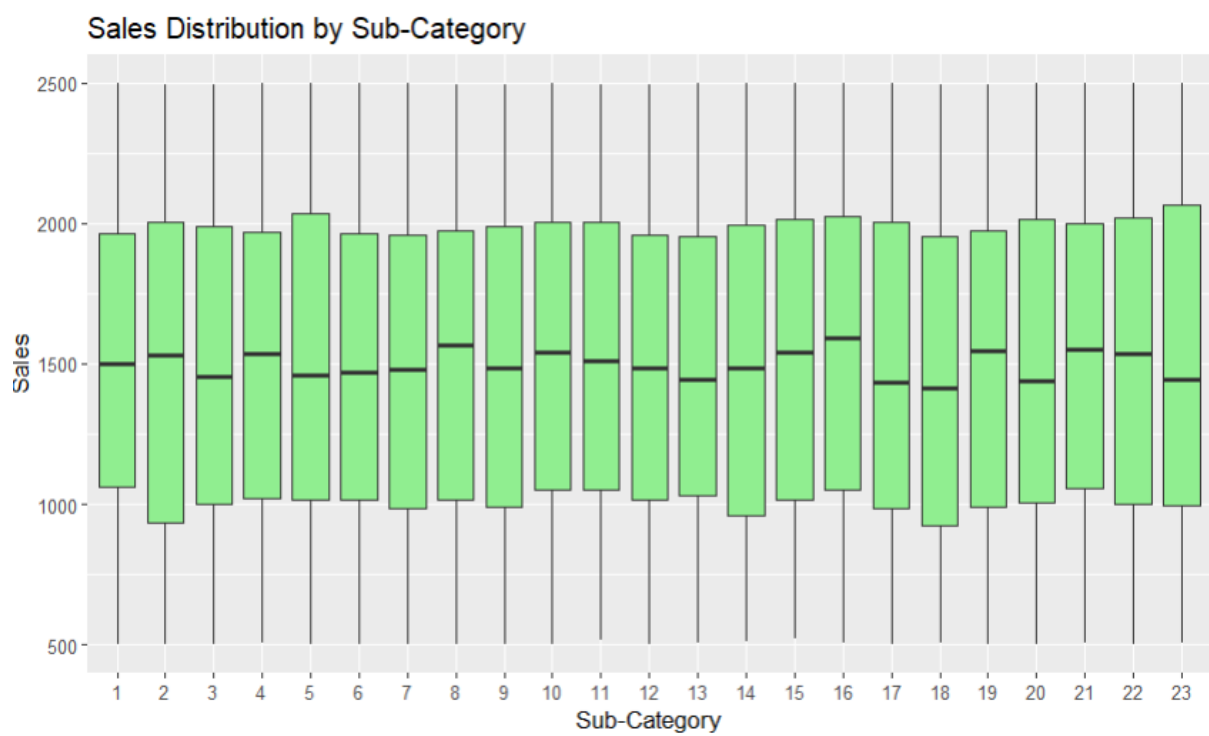
```



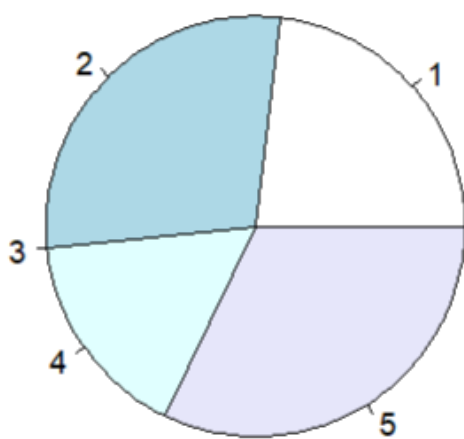
Sales Distribution by Category







Sales Distribution by Region



CONCLUSION

The analysis provided valuable insights into product category performance, revealing high-revenue categories such as electronics and clothing, while identifying underperformers like home goods. By examining sales trends over time, we discovered seasonal variations, highlighting peaks during holidays and the need for timely marketing campaigns. Additionally, potential areas for optimization were identified, including inventory management to align stock with anticipated demand, targeted marketing strategies for both high-performing and underperforming categories, and customer segmentation to enhance engagement. Predictive modeling further enabled forecasting of future sales trends based on historical data, facilitating better resource allocation and risk mitigation. This comprehensive approach ensures informed decision-making and strategic planning for sustained growth in a competitive marketplace.