Retail Analytics Term 2 Project

Spring 2025, University of West Georgia

By

Enita Omuvwie

Table of contents

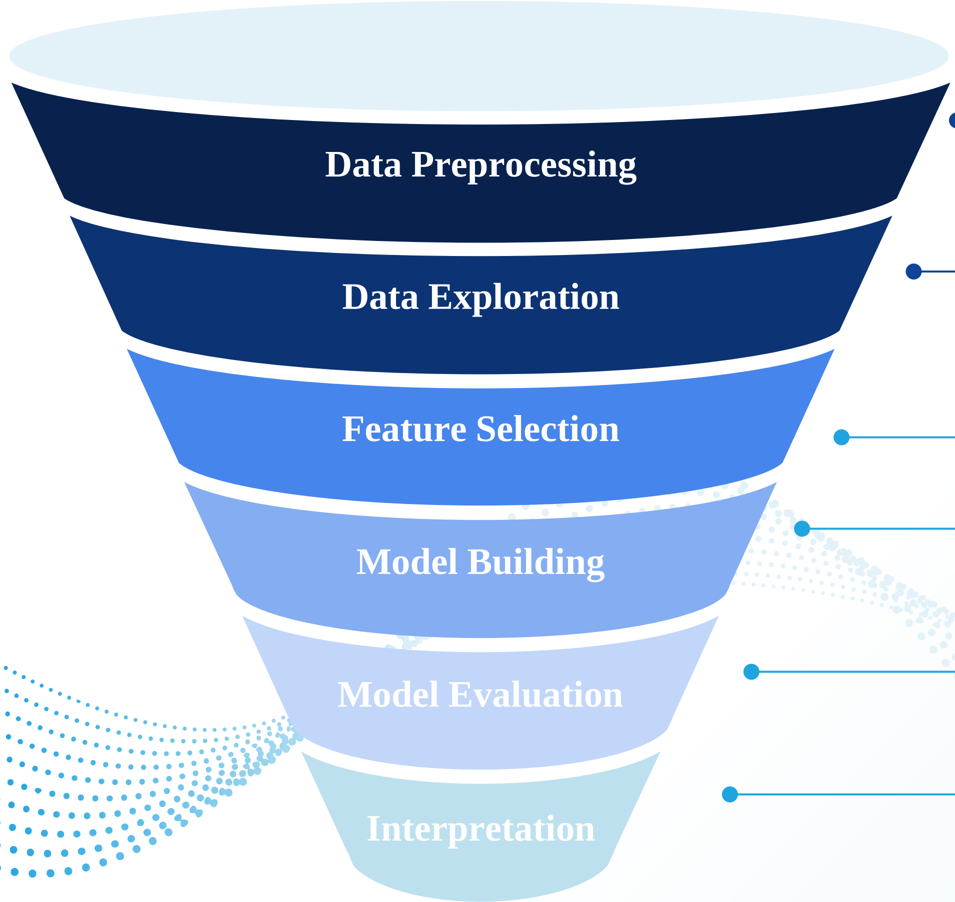
Objective

- Develop a DataFrame for marketing data analysis and prediction
- Conduct exploratory data analysis (EDA) to examine distributions, outliers, and correlations
- Build a linear regression model to identify the best predictor of DEPVAR (Total_Amount)
- Evaluate model accuracy and derive predictive equation for future analysis

Background

- The dataset integrates **Customer, Product, and Transaction** data to form **Marketing_data.csv** for analyzing customer behavior, product performance, and sales trends
- Through **linear regression** and **exploratory data analysis**, we aim to identify key predictors of total sales, detect trends, assess relationships between variables and gain insights into market patterns
- Further analyses like **correlation analysis**, **quartile distribution**, and **outlier detection**, will provide deeper insights to support data driven marketing strategies

Methodology



- Import cleaned dataset
- Check for outliers
- Convert categorical variables to numerical representations
- Compute Descriptive statistics – Quartiles, Mean
- Identify outliers using boxplots and quartile distribution
- Analyze distribution of independent variables
- Correlation analysis using hestmaps to identify relationships
- Cross tabulation for categorical variables with DEPVAR
- Identify the best predictor variables for DEPVAR using correlation analysis
- Perform Linear regression with the best independent variables
- Visualize model results using scatter plots
- Assess the model accuracy using R-squared, Adjusted R-squared, RMSE and MS
- Interpret coefficients to understand the impact of indepenednt variables on sales
- Make data-driven recommendations

Code and Output : Data Preprocessing

Reading in the data, convert date variable and creating the dependent variables

[1]:
```
# Reading in the libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

[2]:
```
# Define column names
column_names = ["Cust_ID", "Cust_Gender", "Cust_Age","Prdct_ID","Prdct_Category","Prdct_Amt","Trnst_ID","Trnst_Date","Prch_Qnty"]

# Reading in the csv file
marketing_df = pd.read_csv('Marketing_data.csv', names=column_names, header = None)
```

[3]:
```
# Changing the datatype of date
marketing_df['Trnst_Date'] = pd.to_datetime(marketing_df['Trnst_Date'])
```

[4]:
```
# Creating the total spent column
marketing_df['DEPVAR'] = marketing_df['Prch_Qnty'] * marketing_df['Prdct_Amt']

print(marketing_df.head())
```

```
   Cust_ID Cust_Gender  Cust_Age  Prdct_ID Prdct_Category  Prdct_Amt  \
0  CUST001           M        34      2551             Be       50.0
1  CUST010           F        52      3671             Cl       50.0
2  CUST100           M        41      2226             El       30.0
3  CUST101           M        32      4424             Cl      300.0
4  CUST102           F        47      3815             Be       25.0

   Trnst_ID Trnst_Date  Prch_Qnty  DEPVAR
0         1 2023-11-24          3   150.0
1        10 2023-10-07          4   200.0
2       100 2023-06-16          1    30.0
3       101 2023-01-29          2   600.0
4       102 2023-04-28          2    50.0
```

Code and Output : Data Preprocessing

Checking the data structure of the variables in the data frame

[5]:
```
# printing the basic structure of the dataframe
print(marketing_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 806 entries, 0 to 805
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Cust_ID         806 non-null    object
 1   Cust_Gender     806 non-null    object
 2   Cust_Age        806 non-null    int64
 3   Prdct_ID        806 non-null    int64
 4   Prdct_Category  806 non-null    object
 5   Prdct_Amt       806 non-null    float64
 6   Trnst_ID        806 non-null    int64
 7   Trnst_Date      806 non-null    datetime64[ns]
 8   Prch_Qnty       806 non-null    int64
 9   DEPVAR          806 non-null    float64
dtypes: datetime64[ns](1), float64(2), int64(4), object(3)
memory usage: 63.1+ KB
None
```

Code and Output : Data Exploration

Calculating the Quartile Distribution and Average of the Dependent Variable

[6]:
```
# Calculating the Quartile distribution
quartiles = marketing_df["DEPVAR"].describe()
print(quartiles)
```

```
count     806.000000
mean      455.973945
std       559.234459
min        25.000000
25%        60.000000
50%       150.000000
75%       900.000000
max      2000.000000
Name: DEPVAR, dtype: float64
```

Code and Output : Data Exploration

Checking for Outliers (no outlier was found) and Average of Total Amount ($455.97)

```
# Identifying Outliers using IQR
Q1 = marketing_df["DEPVAR"].quantile(0.25)
Q3 = marketing_df["DEPVAR"].quantile(0.75)
IQR = Q3 - Q1

# Defining Left & Right Outlier Boundaries
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
print(f"Lower Bound: {lower_bound:.2f}")
print(f"Upper Bound: {upper_bound:.2f}")

# Identifying Outliers
left_outliers = marketing_df[marketing_df["DEPVAR"] < lower_bound]
right_outliers = marketing_df[marketing_df["DEPVAR"] > upper_bound]

# Display Results
print("\nAverage Values:", marketing_df["DEPVAR"].mean())
print("\nLeft Outliers:")
print(left_outliers)
print("\nRight Outliers:")
print(right_outliers)
```

```
Lower Bound: -1200.00
Upper Bound: 2160.00

Average Values: 455.9739454094293

Left Outliers:
Empty DataFrame
Columns: [Cust_ID, Cust_Gender, Cust_Age, Prdct_ID, Prdct_Category, Prdct_Amt, Trnst_ID, Trnst_Date, Prch_Qnty, DEPVAR]
Index: []

Right Outliers:
Empty DataFrame
Columns: [Cust_ID, Cust_Gender, Cust_Age, Prdct_ID, Prdct_Category, Prdct_Amt, Trnst_ID, Trnst_Date, Prch_Qnty, DEPVAR]
Index: []
```

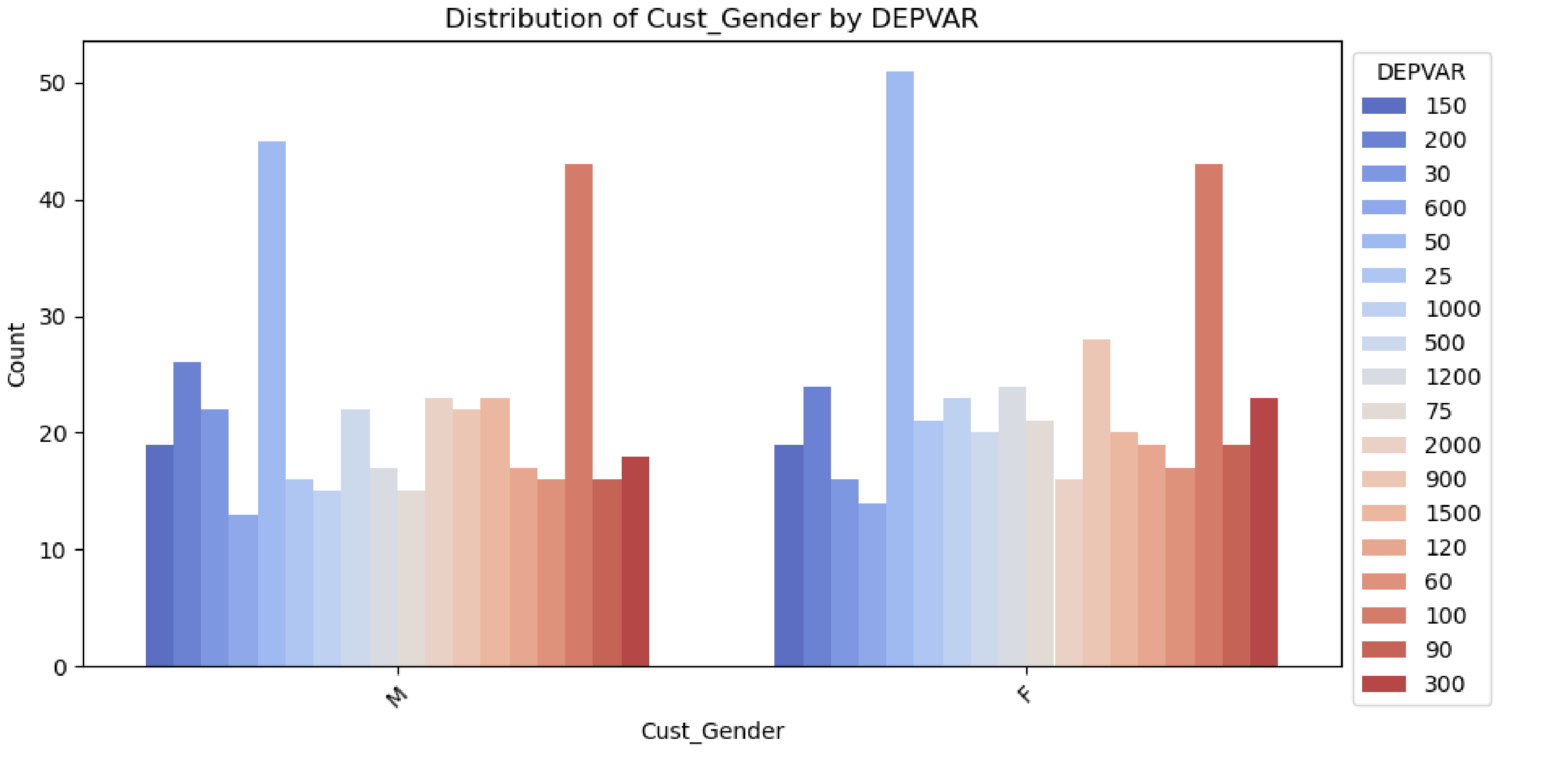
Code and Output : Data Exploration

Distribution of categorical variables against the dependent variable DEPVAR

```
cat_vars = ["Cust_Gender", "Prdct_Category"]  # Add relevant categorical columns

for col in cat_vars:
    plt.figure(figsize=(10, 5))

    # Convert DEPVAR to a categorical variable
    sns.countplot(
        x=marketing_df[col],
        hue=marketing_df["DEPVAR"].round(0).astype(int).astype(str),  # Convert to integer, then string
        palette="coolwarm"
    )

    plt.title(f"Distribution of {col} by DEPVAR")
    plt.xticks(rotation=45)
    plt.xlabel(col)
    plt.ylabel("Count")

    plt.legend(title="DEPVAR", loc="upper left", bbox_to_anchor=(1, 1))  # Ensure legend displays properly
    plt.show()
```

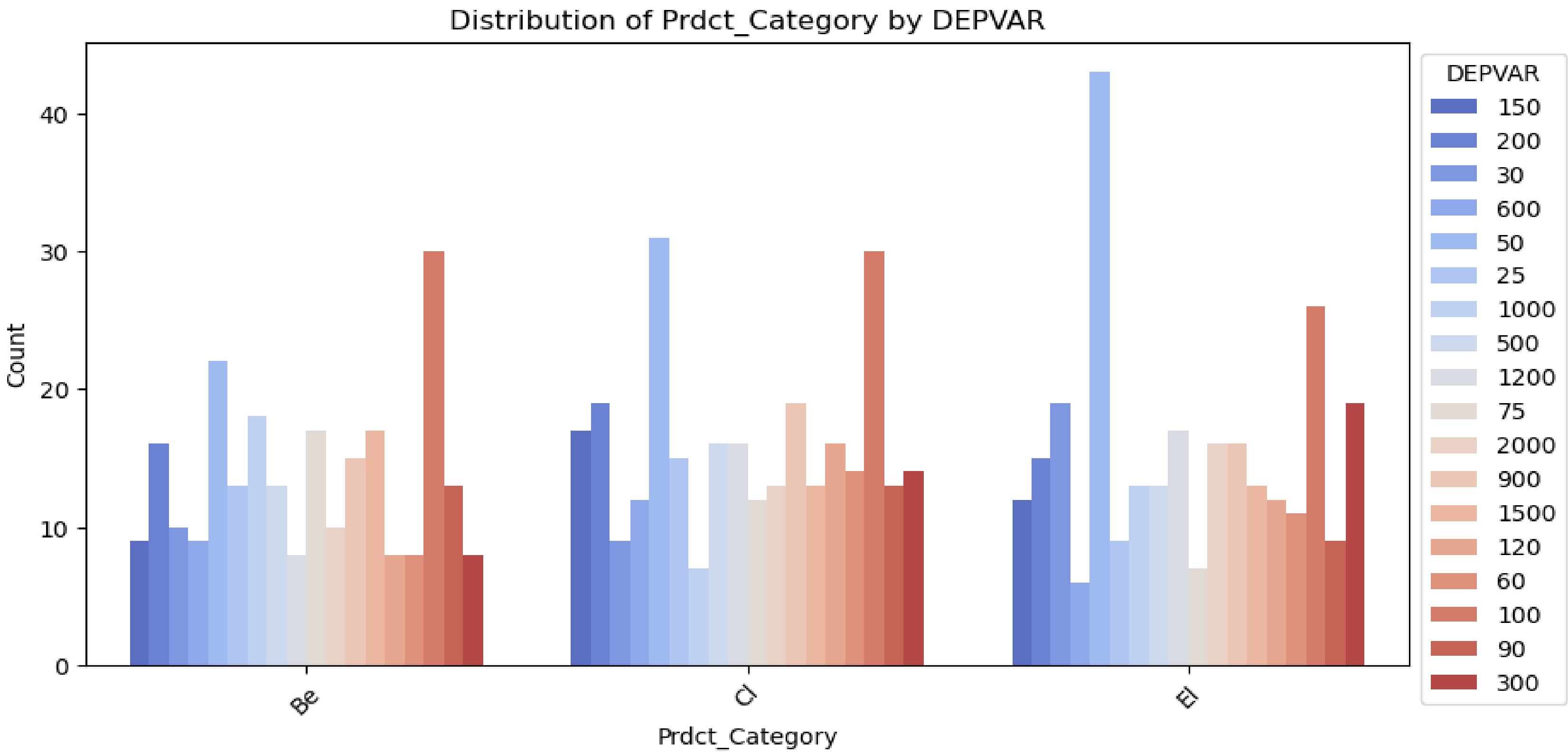
Code and Output : Data Exploration

Distribution of Customer Gender by the dependent variable DEPVAR shows similar patterns for both genders, with peaks at certain DEPVAR values



Code and Output : Data Exploration

Distribution of Product Categories by the dependent variable DEPVAR, with some categories showing higher counts at specific DEPVAR values



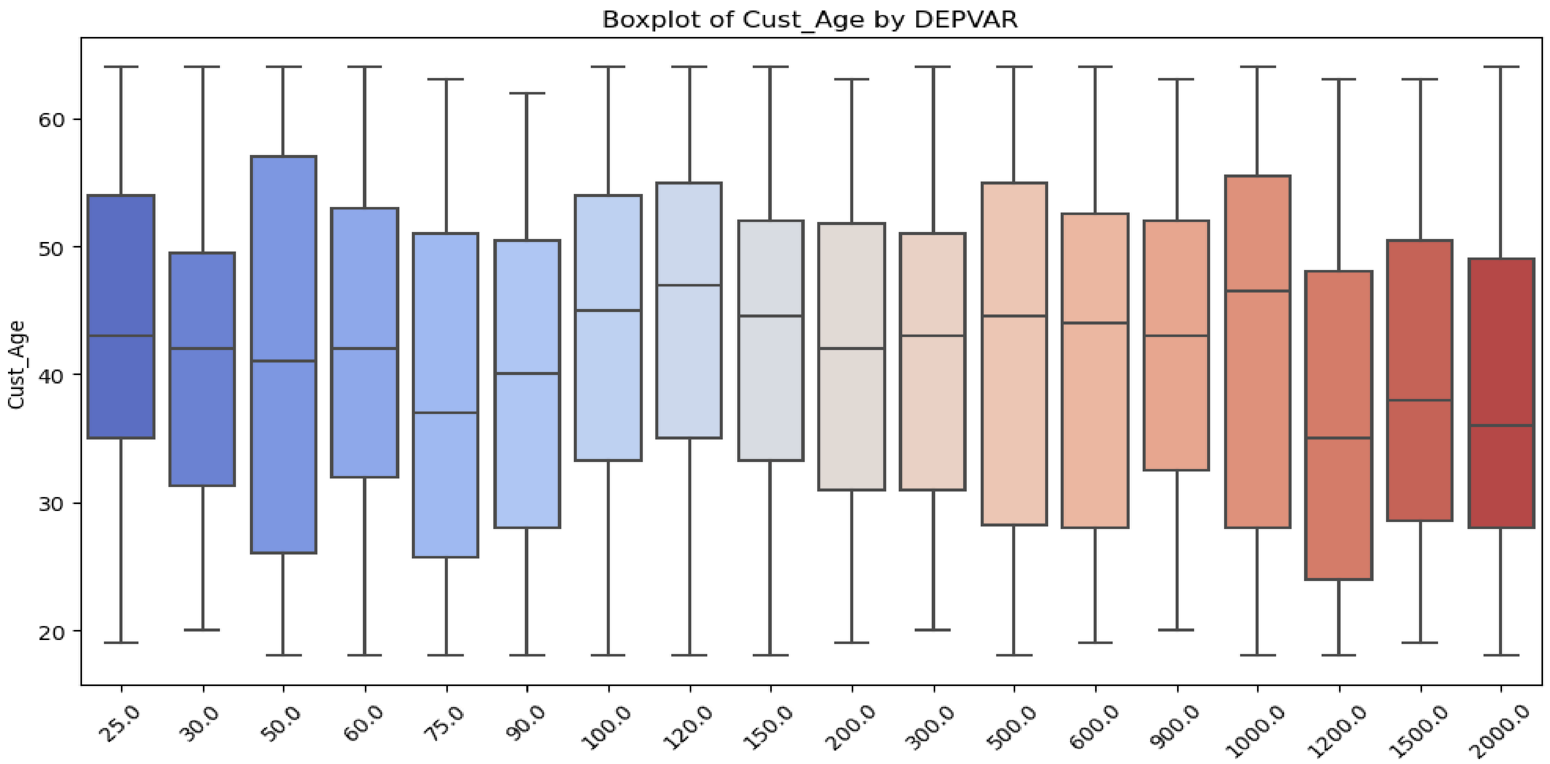
Code and Output : Data Exploration

Distribution of numerical variables against the dependent variable DEPVAR

```
num_vars = ["Cust_Age", "Prdct_Amt", "Prch_Qnty"]

for col in num_vars:
    plt.figure(figsize=(12, 6))  # Make the figure wider
    ax = sns.boxplot(x=marketing_df["DEPVAR"], y=marketing_df[col], palette="coolwarm")

    # Adjust x-axis
    plt.xticks(rotation=45)  # Rotate labels for better readability
    ax.set_xticks(ax.get_xticks())  # Ensure proper spacing
    plt.xlabel("DEPVAR")  # Clear x-axis label
    plt.ylabel(col)  # Clear y-axis label
    plt.title(f"Boxplot of {col} by DEPVAR")

    plt.show()
```

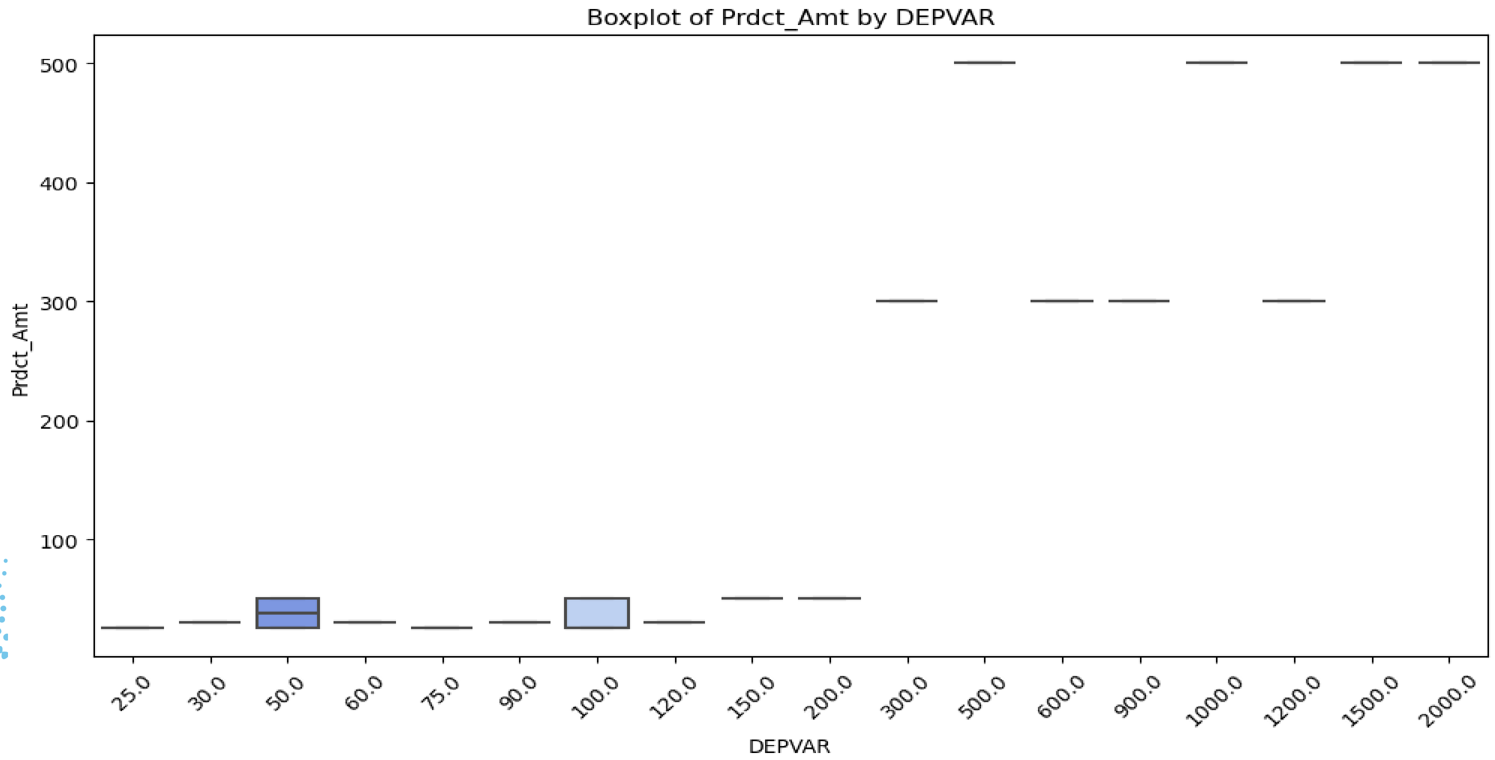
Code and Output : Data Exploration

Customer Age against the dependent variable DEPVAR, with median ages between 30-50 and variation in spread



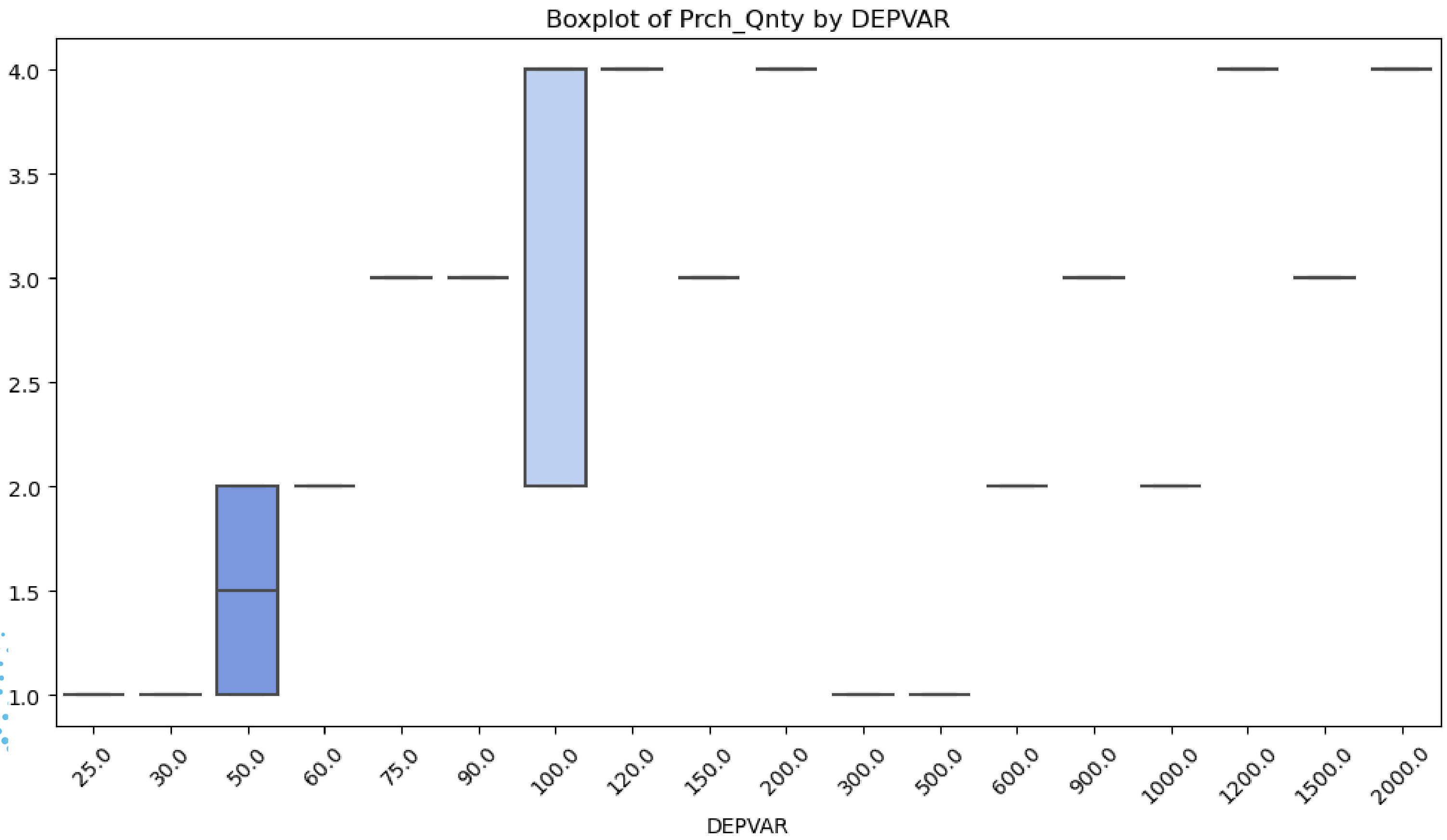
Code and Output : Data Exploration

Product Amount against the dependent variable DEPVAR, it has significant outliers at higher levels of DEPVAR and more concentrated at lower amounts



Code and Output : Data Exploration

Product Quantity against the dependent variable DEPVAR, some categories have higher median quantities

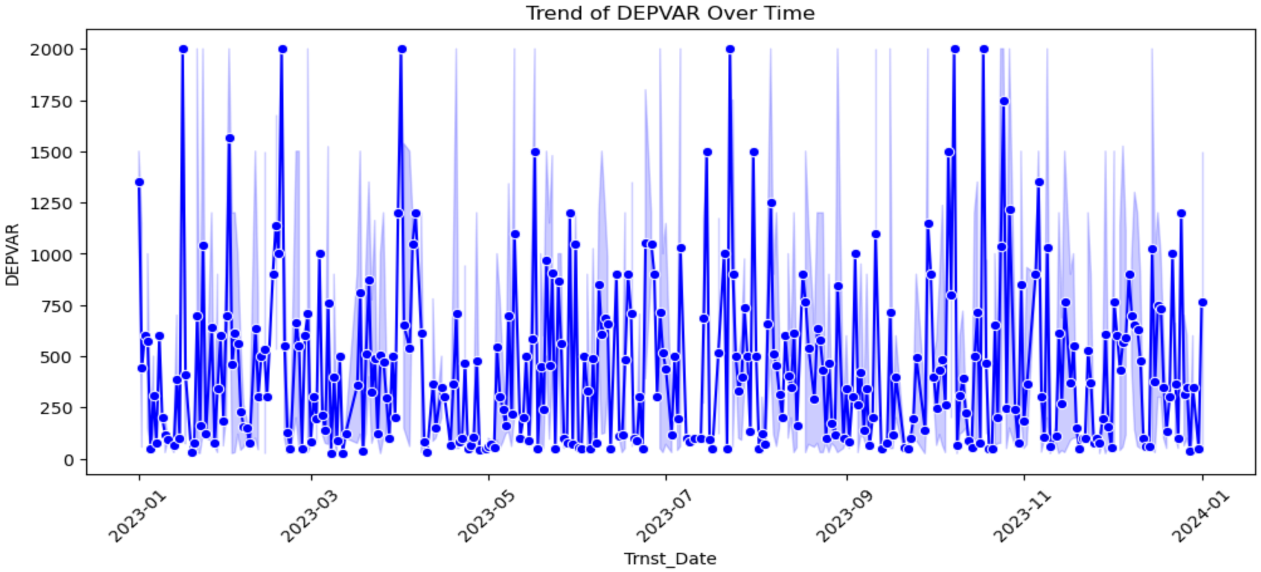


Code and Output : Data Exploration

Trend of the dependent variable DEPVAR over Transaction Time showing frequent fluctuations with several peaks through the year

```
plt.figure(figsize=(12, 5))
sns.lineplot(data=marketing_df, x="Trnst_Date", y="DEPVAR", marker="o", color="blue")
plt.title("Trend of DEPVAR Over Time")
plt.xticks(rotation=45)
plt.show()
```

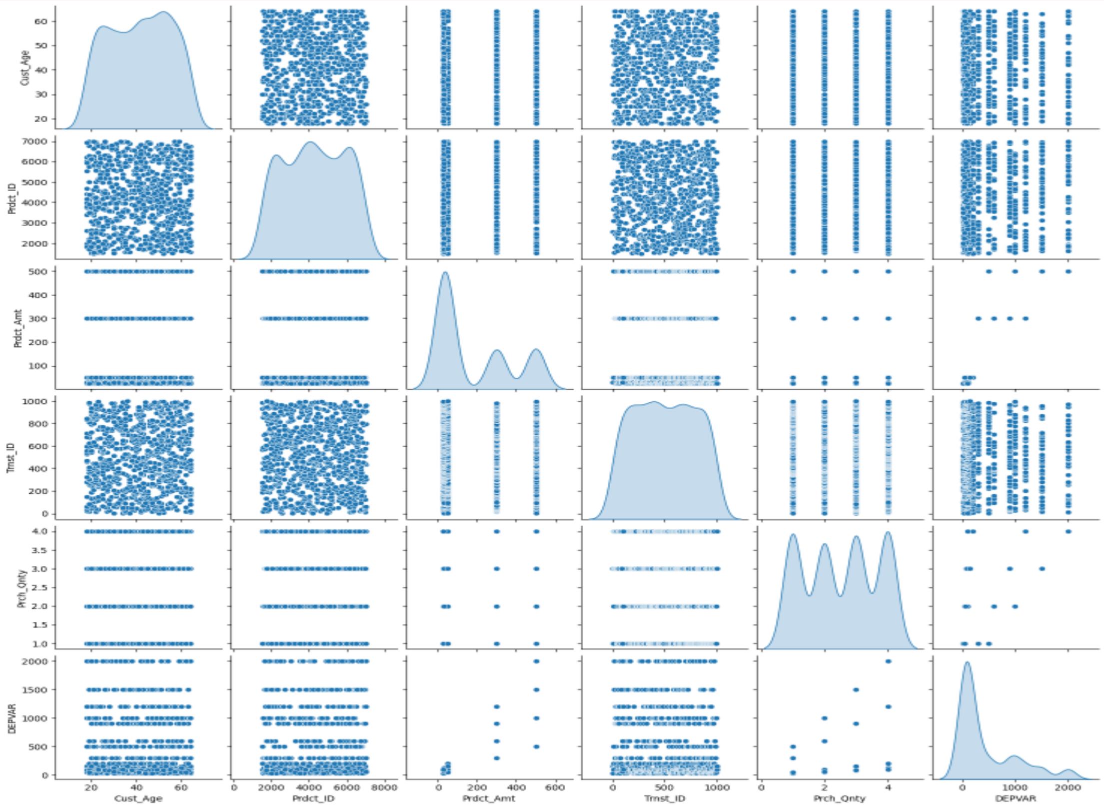
```
/opt/conda/envs/anaconda-2024.02-py310/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/envs/anaconda-2024.02-py310/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a
  with pd.option_context('mode.use_inf_as_na', True):
```



Code and Output : Data Exploration

Pair plot Analysis of varied distributions and relationships among variables

```
# Pairplot Analysis
sns.pairplot(marketing_df, diag_kind="kde")
plt.show()
```



Code and Output : Data Exploration

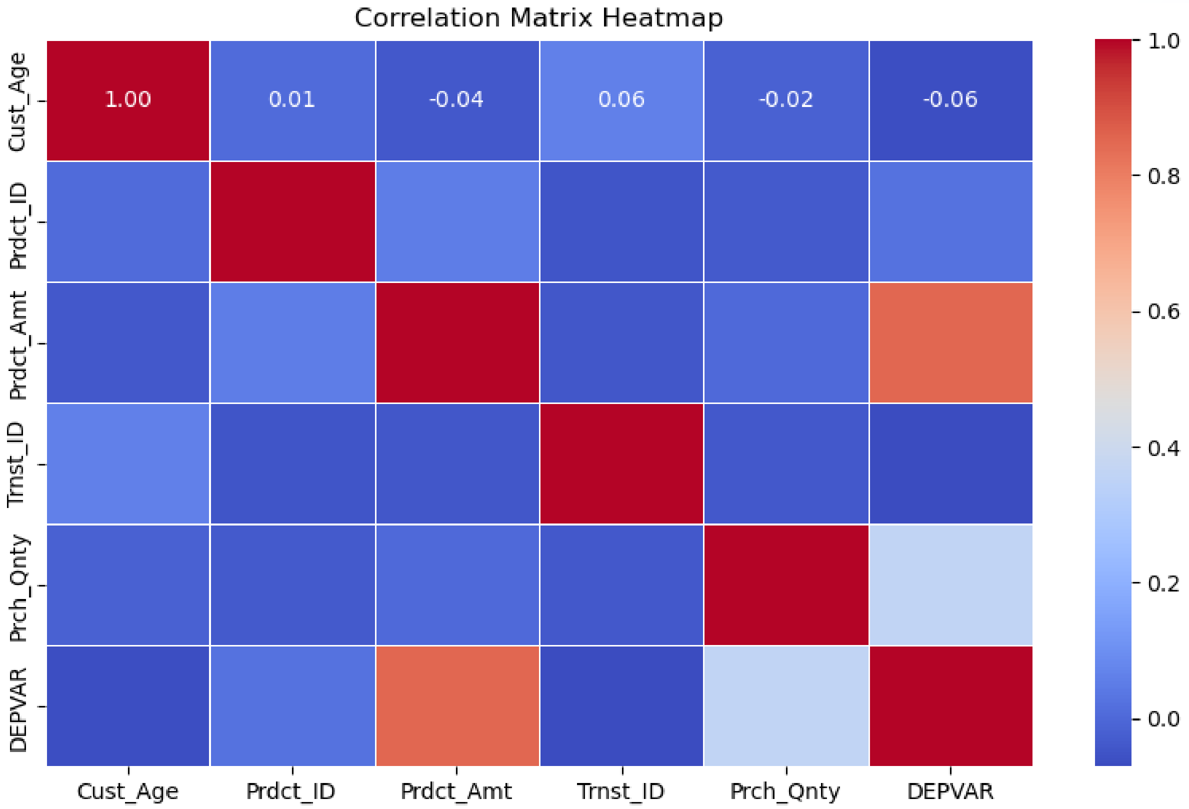
Correlation Analysis of numerical variables

```
# Correlation Analysis
# Select only numeric columns from the dataframe
numeric_df = marketing_df.select_dtypes(include=[np.number])

# Calculate the correlation matrix for the numeric columns
correlation_matrix = numeric_df.corr()

# Print or display the correlation matrix
print(correlation_matrix)

# Set up the matplotlib figure
plt.figure(figsize=(10, 6))

# Create the heatmap with annotations
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)

# Set the title
plt.title('Correlation Matrix Heatmap')

# Display the plot
plt.show()
```

```
            Cust_Age  Prdct_ID  Prdct_Amt  Trnst_ID  Prch_Qnty    DEPVAR
Cust_Age    1.000000  0.005015  -0.038559  0.061027  -0.016724 -0.064214
Prdct_ID    0.005015  1.000000   0.050694 -0.047410  -0.033546  0.025219
Prdct_Amt  -0.038559  0.050694   1.000000 -0.045604   0.001142  0.850036
Trnst_ID    0.061027 -0.047410  -0.045604  1.000000  -0.041468 -0.070772
Prch_Qnty  -0.016724 -0.033546   0.001142 -0.041468   1.000000  0.363580
DEPVAR     -0.064214  0.025219   0.850036 -0.070772   0.363580  1.000000
```

Code and Output : Data Exploration

Correlation Analysis using Heatmap- with weak correlations between most variables, with some moderate correlations involving DEPVAR



Code and Output : Data Exploration

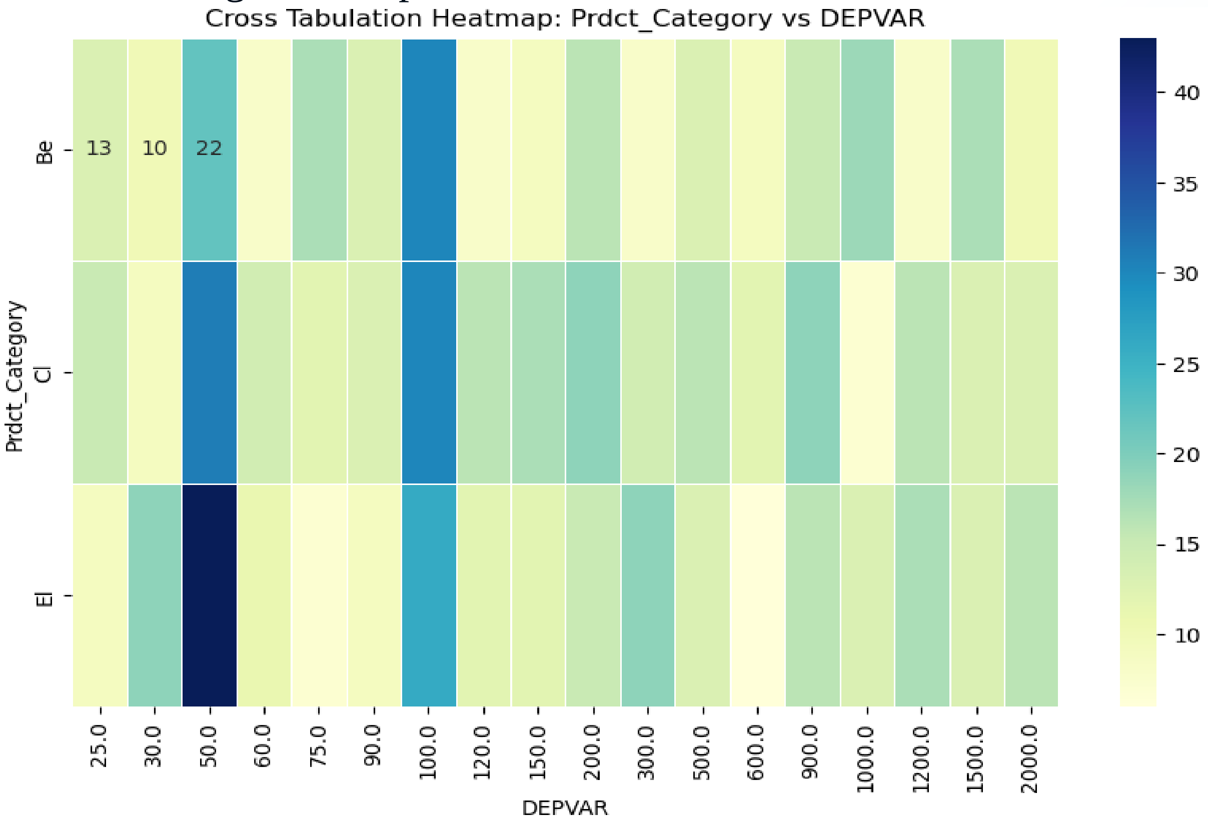
Cross frequency analysis for each product flag and dependent variable DEPVAR

```
# Cross-frequency analysis for each product flag and DEPVAR
if "Prdct_Category" in marketing_df.columns:
    cross_tab = pd.crosstab(marketing_df["Prdct_Category"], marketing_df["DEPVAR"])
    print(cross_tab)

# Plotting the heatmap
    plt.figure(figsize=(10, 6))
    sns.heatmap(cross_tab, annot=True, cmap='YlGnBu', fmt='d', linewidths=0.5)

    # Adding title and labels
    plt.title('Cross Tabulation Heatmap: Prdct_Category vs DEPVAR')
    plt.xlabel('DEPVAR')
    plt.ylabel('Prdct_Category')

    # Show the plot
    plt.show()
```

```
DEPVAR          25.0   30.0   50.0   60.0   75.0   90.0   100.0  \
Prdct_Category
Be                13     10     22      8     17     13      30
Cl                15      9     31     14     12     13      30
El                 9     19     43     11      7      9      26

DEPVAR          120.0  150.0  200.0  300.0  500.0  600.0  900.0  \
Prdct_Category
Be                 8      9     16      8     13      9      15
Cl                16     17     19     14     16     12      19
El                12     12     15     19     13      6      16

DEPVAR          1000.0 1200.0 1500.0 2000.0
Prdct_Category
Be                18      8     17     10
Cl                 7     16     13     13
El                13     17     13     16
```

Code and Output : Data Exploration

Cross frequency analysis for each product category and dependent variable DEPVAR – highlighting the prevalence of some categories at specific values



Code and Output : Model Building and Evaluation

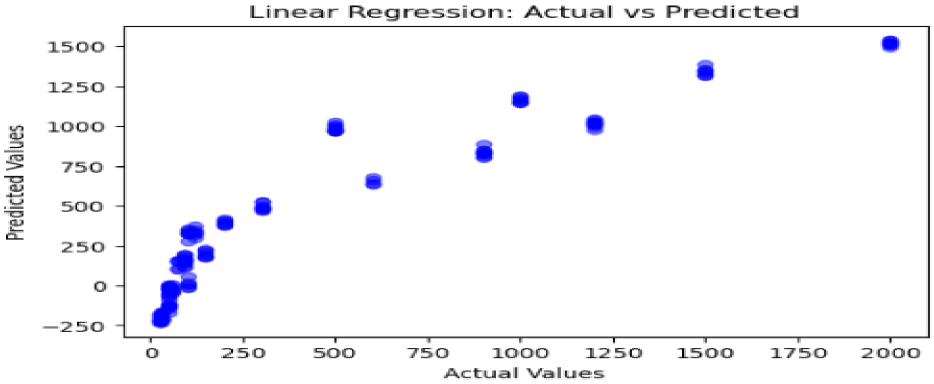
[14]:
```
# Convert categorical variables to dummies
dummies = pd.get_dummies(
    marketing_df[cat_vars],   # Select only specified categorical columns
    prefix=cat_vars,          # Add prefixes like "Cust_Gender_Male"
    drop_first=True           # Avoid dummy variable trap
)
```

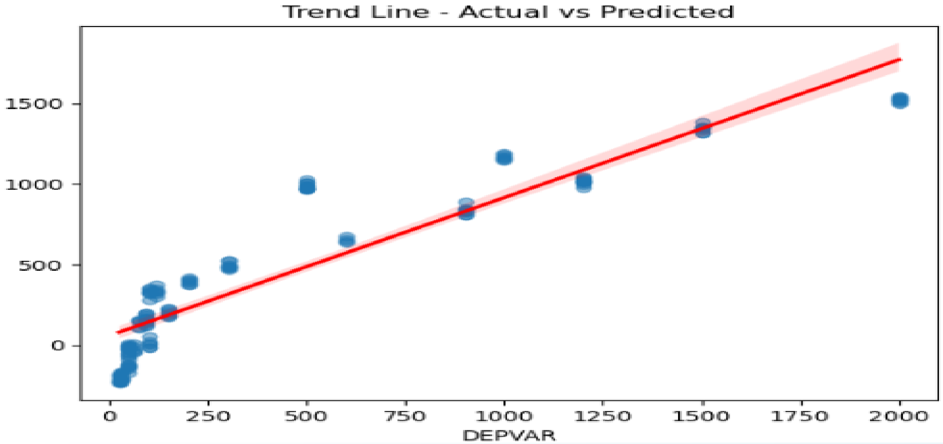
Linear Regression

[15]:
```
# Linear Regression Model
X = pd.concat([
    marketing_df[num_vars],
    dummies
], axis=1)
y = marketing_df['DEPVAR']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training
model = LinearRegression()
model.fit(X_train, y_train)

# Predictions
y_pred = model.predict(X_test)

# Visualization of the Model
plt.figure(figsize=(6, 4))
plt.scatter(y_test, y_pred, alpha=0.5, color="blue")
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.title("Linear Regression: Actual vs Predicted")
plt.show()

# Accuracy: Scatter plot with trend line
sns.regplot(x=y_test, y=y_pred, scatter_kws={"alpha": 0.5}, line_kws={"color": "red"})
plt.title("Trend Line - Actual vs Predicted")
plt.show()

# Model Equation
coefficients = dict(zip(X.columns, model.coef_))
print(f"Model Equation: y = {model.intercept_:.2f} + " + " + ".join([f"{coeff:.2f}*{var}" for var, coeff in coefficients.items()]))


# Model Evaluation
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse:.2f}")

# RMSE (Root Mean Squared Error)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print(f"RMSE: {rmse:.2f}")

r2 = r2_score(y_test, y_pred)
print(f"R-squared: {r2:.2f}")

# Adjusted R-squared
n = len(y)  # Number of observations
p = X.shape[1]  # Number of predictors
adj_r2 = 1 - (1 - r2) * (n - 1) / (n - p - 1)
print(f"Adjusted R-squared: {adj_r2:.2f}")
```

Code and Output : Model Building and Evaluation

The linear regression plot shows strong positive correlation between actual and predicted values, with the trend line closely aligned with data points





Code and Output: Model Interpretation

Model Equation:

y = -403.22 + -1.18***Cust_Age** + 2.52***Prdct_Amt** + 175.09***Prch_Qnty** + 12.39***Cust_Gender_M** + -8.15***Prdct_Category_Cl** + 21.72***Prdct_Category_El**

The model predicts the dependent variable (DEPVAR-Total Amount) using a linear combination of some features

- **Cust_Age** : Has a small negative impact on y hence highlighted in red
- **Prdct_Amt** : Has positive influence on y indicating higher amount leads to higher predictions in the Total Amount-DEPVAR hence highlighted in green
- **Prch_Qnty**: Has a strong impact on DEPVAR with larger quantity having a significant effect on y
- **Cust_Gender_M**: Positive coefficient indicating male customers have higher predicted DEPVAR values
- **Prdct_Category_Cl**: Product category Clothing has a negative impact on DEPVAR
- **Prdct_Category_El** : Product category Electronic has a positive impact on DEPVAR

Code and Output: Model Interpretation

Mean Squared Error (MSE): A value of **51102.46** indicates the averaged squared difference between actual and predicted values

Root Mean Squared Error (RMSE): At **226.06**, provides a measure of the average prediction error in same units as y-DEPVAR

R-squared: An R-squared of **0.84** suggests that 84% of the variance in the dependent variable is explained by the model

Adjusted R-squared: Also **0.84**, indicates the model's explanatory power remains strong even after adjusting the number of predictors

Conclusion

- **Data Distribution:** Data shows varied distribution across both categorical and numerical variables with peaks and patterns
- **Trends Over Time:** DEPVAR shows a fluctuation over time with significant peak periods
- **Correlations:** Most variables have weak correlations, but some had moderate correlations involving DEPVAR
- **Model Performance:** The linear regression model performs well, explaining significant portion of the variance(**R-squared = 0.84**) and a prediction error (**RMSE=226.06**)
- **Key Influencers:** Purchase quantity, product amount, customer gender-male and electronic product category have a strong predictive impact on the DEPVAR and customer age has minimal negative impact on DEPVAR

Thank you

Any questions?

You can find me at

- **eomuvwi1@my.westga.edu**