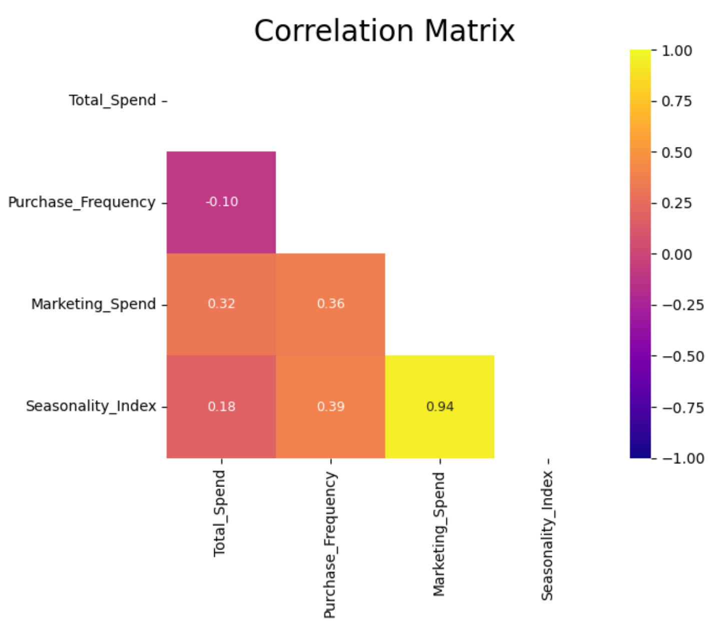
Week 3 - Advanced Data Analysis Techniques and Business Insights

1. Data Loading and Initial Cleaning

- Loaded CSV file: raw_sales_data.csv
- Inspected shape, columns, dtypes, nulls, and duplicates
- Cleaned data:
- ✓ Filled Seasonality_Index missing values with median [Found 1 missing value]
- ✓ Normalized Churned column (Y/N/y/n → Yes/No)
- Subset sales-related columns for correlation analysis

2. Exploratory Data Analysis (EDA)

- Correlation HeatMap



Observations:

➢ Potential Multicollinearity: Strong correlation between Marketing_Spend and Seasonality_Index (**0.94**)

3. **Outlier Detection & Removal**:

➢ Used **Z-score filtering** (<3)

➢ Reduced data size from original to outlier-free version [Rows: 24 to 22]

4. **Class Imbalance Check**
   - Churned distribution showed moderate class imbalance
   - Yes: 13, No: 11

5. **Predictive Modelling**
   - Linear Regression – Total Spend Prediction

```python
# Prepare features and target
X = filtered_data[['Marketing_Spend', 'Seasonality_Index']]
y = filtered_data['Total_Spend']

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```
[25] ✓ 0.0s

```python
# Used StandardScaler to scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```
[26] ✓ 0.0s

```python
# Linear Regression Model
lr = LinearRegression()
lr.fit(X_train_scaled, y_train)

# Predict
y_pred = lr.predict(X_test_scaled)

# Evaluate
rmse = np.sqrt(root_mean_squared_error(y_test, y_pred))
print(f"RMSE: {rmse:.2f}")
print(f"R^2 Score: {r2_score(y_test, y_pred):.2f}")

print(f"Regression Co-efficients:", lr.coef_, lr.intercept_)
```
[27] ✓ 0.0s

```
RMSE: 21.69
R^2 Score: 0.86
Regression Co-efficients: [924.44038178 249.20229739] 3894.1176470588234
```

Peformance:

- R^2 = 86% → strong explanatory power in Total Spend
- RMSE = 21.69 → low prediction error
- Marketing_Spend had higher impact than Seasonality_Index

- Logistic Regression – Churn Prediction

```python
# Encode target variable
le = LabelEncoder()
raw_sales_data['Churned'] = le.fit_transform(raw_sales_data['Churned'])  # 'Yes'=1, 'No'=0

# Features and target
features = ['Marketing_Spend', 'Seasonality_Index', 'Purchase_Frequency']
X = raw_sales_data[features]
y = raw_sales_data['Churned']

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

# Scale features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Logistic Regression
logreg = LogisticRegression()
logreg.fit(X_train_scaled, y_train)

# Predictions
y_pred = logreg.predict(X_test_scaled)

# Evaluation
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}")
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```
[28] ✓ 0.0s

Output:

```
Accuracy: 0.80
Confusion Matrix:
 [[3 0]
 [1 1]]
Classification Report:
               precision    recall  f1-score   support

           0       0.75      1.00      0.86         3
           1       1.00      0.50      0.67         2

    accuracy                           0.80         5
   macro avg       0.88      0.75      0.76         5
weighted avg       0.85      0.80      0.78         5
```

Performance:

- Accuracy: 80%
- High precision and recall; solid model for churn prediction

6. Statistical Analysis for Business Insights

H_0 (Null Hypothesis): All regions have the same average Total Spend.
H_1 (Alternative Hypothesis): At least one region's average Total Spend is different.

- ANOVA (Across Regions):
  - Compared the average Total Spend across multiple regions (North, South, East, West).
  - p-value > 0.05 → **Can not Reject H_0** → There is no significant difference in sales across regions.

```
# Anova test to perform sales analysis over different regions

region_1 = raw_sales_data[raw_sales_data['Region'] == 'North']['Total_Spend']
region_2 = raw_sales_data[raw_sales_data['Region'] == 'South']['Total_Spend']
region_3 = raw_sales_data[raw_sales_data['Region'] == 'East']['Total_Spend']
region_4 = raw_sales_data[raw_sales_data['Region'] == 'West']['Total_Spend']
f_statistic, p_value = f_oneway(region_1, region_2, region_3, region_4)
print(f"F-statistic: {f_statistic:.2f}, p-value: {p_value:.4f}")
if p_value < 0.05:
    print("There are significant differences in Total Spend across regions.")
else:
    print("No significant differences in Total Spend across regions.")
[30]  ✓ 0.0s

... F-statistic: 1.36, p-value: 0.2822
    No significant differences in Total Spend across regions.
```

- Hypothesis Testing: T-test (Effect of Promotions):

  H_0 (Null Hypothesis): Promotions **do not** affect Total Spend
  H_1 (Alternative Hypothesis): Promotions **increase** Total Spend

- Promo group: Customers with above-median Marketing Spend
- Non-promo group: Customers with median or below Marketing Spend
- P-value > 0.05 → **Can not Reject H_0**

```
# Split based on Marketing Spend (above vs below/equal to median)
median_spend = raw_sales_data['Marketing_Spend'].median()
promo_group = raw_sales_data[raw_sales_data['Marketing_Spend'] > median_spend]['Total_Spend']
non_promo_group = raw_sales_data[raw_sales_data['Marketing_Spend'] <= median_spend]['Total_Spend']

# T-test
t_stat, p_val = ttest_ind(promo_group, non_promo_group)

print(f"T-statistic: {t_stat:.2f}")
print(f"P-value: {p_val:.4f}")
if p_val < 0.05:
    print("Promotions significantly affect total spend.")
else:
    print("No significant impact of promotions on total spend. Null hypothesis cannot be rejected.")
[31]  ✓ 0.0s

... T-statistic: 1.42
    P-value: 0.1683
    No significant impact of promotions on total spend. Null hypothesis cannot be rejected.
```

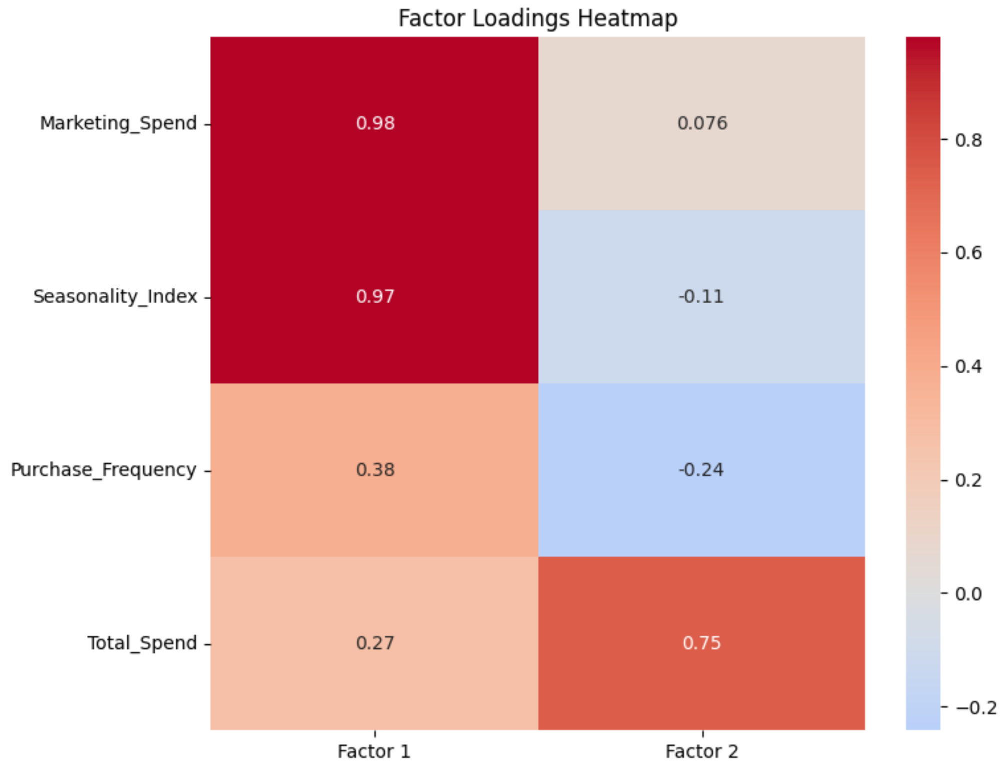
➢ **Factor Analysis:**

```python
# Choose relevant features
features = ['Marketing_Spend', 'Seasonality_Index', 'Purchase_Frequency', 'Total_Spend']
X = raw_sales_data[features]

# Standardize
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Apply Factor Analysis
fa = FactorAnalysis(n_components=2, random_state=42)
X_factors = fa.fit_transform(X_scaled)

# Check loadings
factor_loadings = pd.DataFrame(fa.components_.T, columns=['Factor 1', 'Factor 2'], index=features)
print(factor_loadings)
```

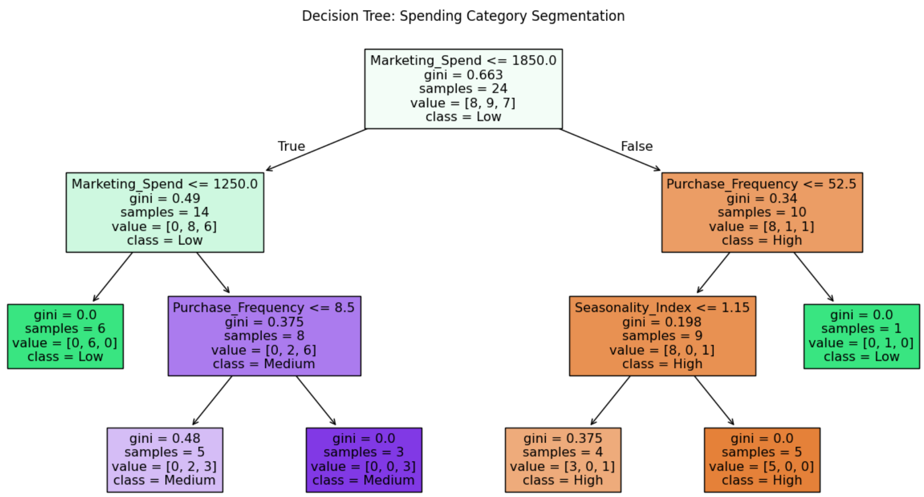


<u>**Observations:**</u>

- Factor 1: High loadings on Marketing_Spend and Seasonality_Index
  → Represents Marketing Influence
- Factor 2: High loading on Total_Spend
  → Represents Customer Spending Power

7. Machine Learning for Customer Segmentation

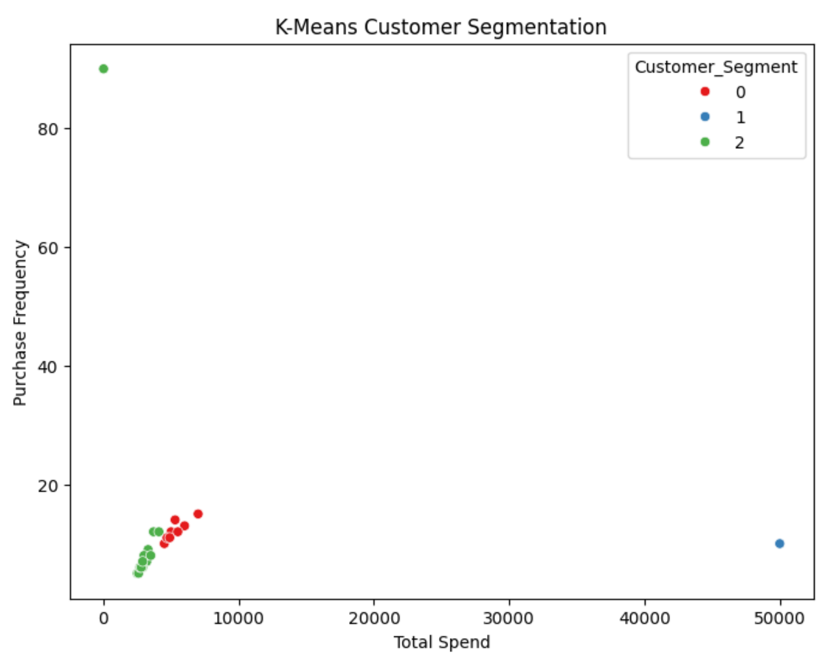
- Decision Tree Classifier



<u>Observations:</u>

- Customers with low marketing spend (≤ 1250) are mostly Low spenders.
- High spenders often:
  - Have high marketing spend (> 1850),
  - Purchase frequently, and Vary by seasonality index.

- K-Means Clustering



Observations:

- Random Forests

```python
# Prepare features/target
features = ['Marketing_Spend', 'Seasonality_Index', 'Purchase_Frequency']
X = raw_sales_data[features]
y = raw_sales_data['Churned']  # 0/1 encoded

# Split
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, random_state=42)

# Train Random Forest
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Predict & Evaluate
y_pred_rf = rf.predict(X_test)
print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
print("Classification Report:\n", classification_report(y_test, y_pred_rf))
```

```
[41]  ✓ 0.1s

Random Forest Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         3

    accuracy                           1.00         6
   macro avg       1.00      1.00      1.00         6
weighted avg       1.00      1.00      1.00         6
```

- XGBoost

```python
xgb = XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_state=42)
xgb.fit(X_train, y_train)          Chat (CTRL + I) / Share (CTRL + L)

# Predict & Evaluate
y_pred_xgb = xgb.predict(X_test)
print("XGBoost Accuracy:", accuracy_score(y_test, y_pred_xgb))
print("Classification Report:\n", classification_report(y_test, y_pred_xgb))
```

```
[40]  ✓ 0.1s

XGBoost Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         3

    accuracy                           1.00         6
   macro avg       1.00      1.00      1.00         6
weighted avg       1.00      1.00      1.00         6
```

Summary and Business recommendations

This project involved analyzing customer sales data to understand spending behavior, churn patterns, and segment customers for targeted marketing.

After data cleaning and outlier removal, key insights from EDA revealed a strong correlation between marketing spend and seasonality.

Predictive models, including linear and logistic regression, showed strong performance, with marketing spend being a major driver of total spend and churn. Statistical tests (ANOVA, t-test) indicated no significant regional or promotional effects, though trends suggested promotional influence. Decision trees and K-means clustering effectively segmented customers into actionable groups. High-accuracy churn models (Random Forest, XGBoost) further enabled predictive targeting.

Recommendations:

1. Collect more data for better predictions since the test data was quite small.
2. Use discounts to increase spending for price-sensitive customers.
3. Since marketing spend is a key driver, focus on it on regions and seasons where it most impacts purchase decisions.