Advanced Data Analysis Techniques and Business Insights

The dataset contains sales transaction records enriched with fields such as:

- Customer_ID, Product_ID, Date, Total_Spend, Marketing_Spend, Region, Promotion, and Churn_Status.

The objective is to extract actionable business insights, forecast sales trends, and understand customer behavior using data science techniques.

	Customer_ID	Customer_Name	Region	Total_Spend	Purchase_Frequency	Marketing_Spend	Seasonality_Index	Churned
0	101	John Doe	North	5000	12	2000	1.2	No
1	102	Jane Smith	South	3000	8	1500	1.0	Yes
2	103	Sam Brown	East	4500	10	1800	1.1	No
3	104	Linda Johnson	West	2500	5	1000	0.9	Yes
4	105	Michael Lee	North	7000	15	2500	1.3	No

Phase 1: Data Cleaning and Preprocessing

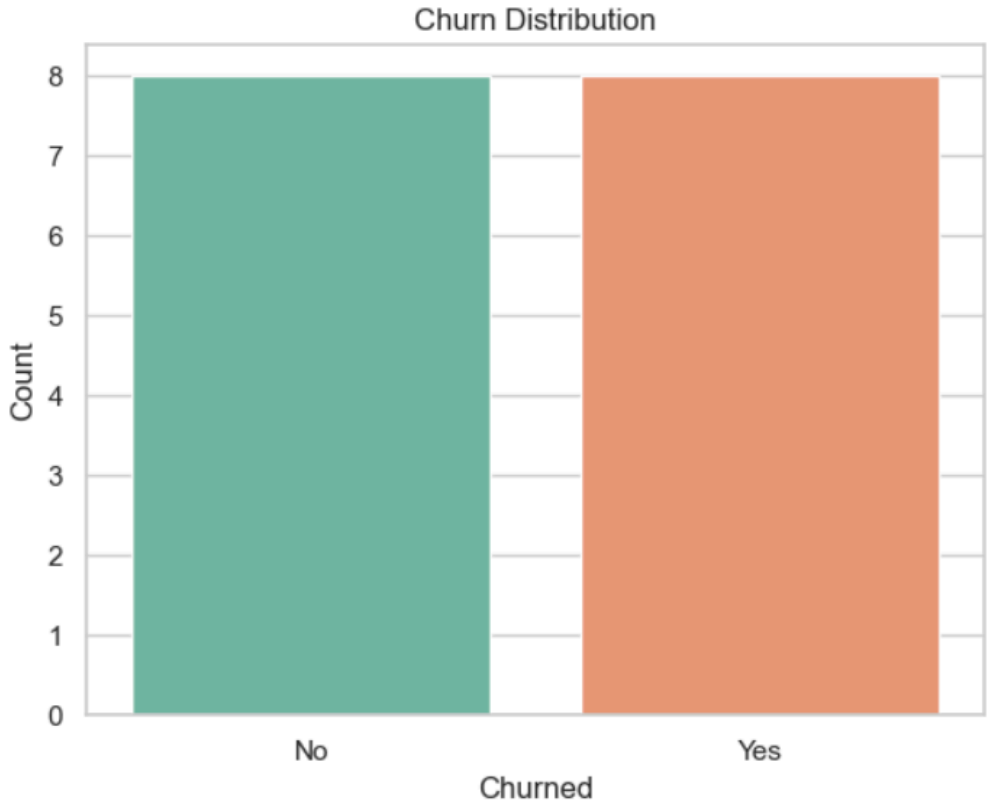
First, the relevant data is collected, i.e., pre-processing the data. This involved preparing the data for analysis by cleaning it, dealing with missing values, and transforming it into an appropriate format. I have checked the missing values, duplicates, data types, and standardized categorical columns. To find any outliers or anomalies in the data that might have an impact on the success of the model, I've also conducted exploratory data analysis.

The resulting dataset had 16 rows and 8 columns.

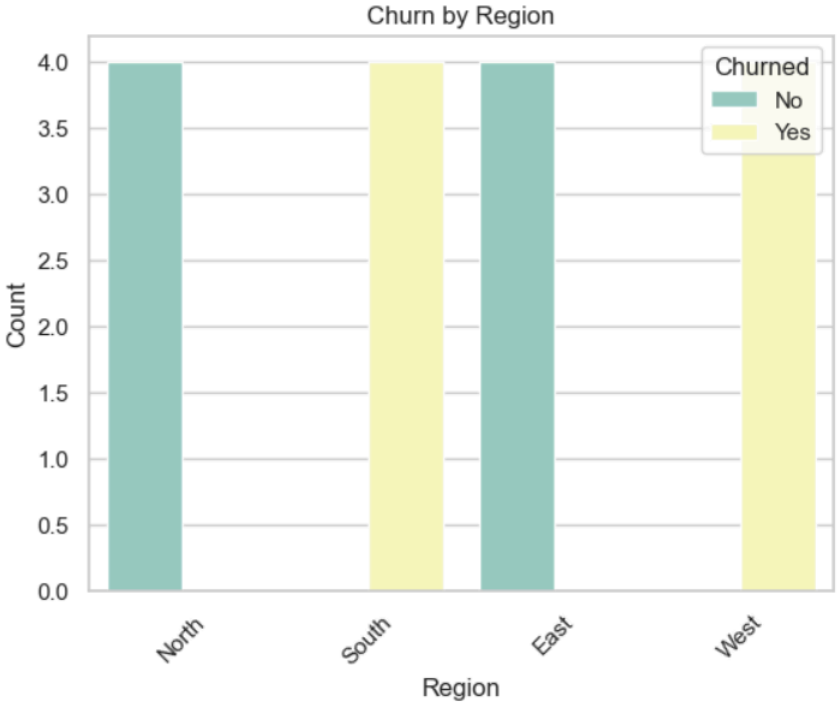
```
Data shape: (16, 8)
Column data types:
Customer_ID              int64
Customer_Name           object
Region                  object
Total_Spend              int64
Purchase_Frequency       int64
Marketing_Spend          int64
Seasonality_Index      float64
Churned               category
dtype: object
First few cleaned records:
   Customer_ID  Customer_Name Region  Total_Spend  Purchase_Frequency  \
0          101       John Doe  North         5000                  12
1          102     Jane Smith  South         3000                   8
2          103      Sam Brown   East         4500                  10
3          104  Linda Johnson   West         2500                   5
4          105    Michael Lee  North         7000                  15

   Marketing_Spend  Seasonality_Index Churned
0             2000                1.2      No
1             1500                1.0     Yes
2             1800                1.1      No
3             1000                0.9     Yes
4             2500                1.3      No
```

Phase 2: Exploratory Data Analysis (EDA)

Visualize Target Variable (Churned)

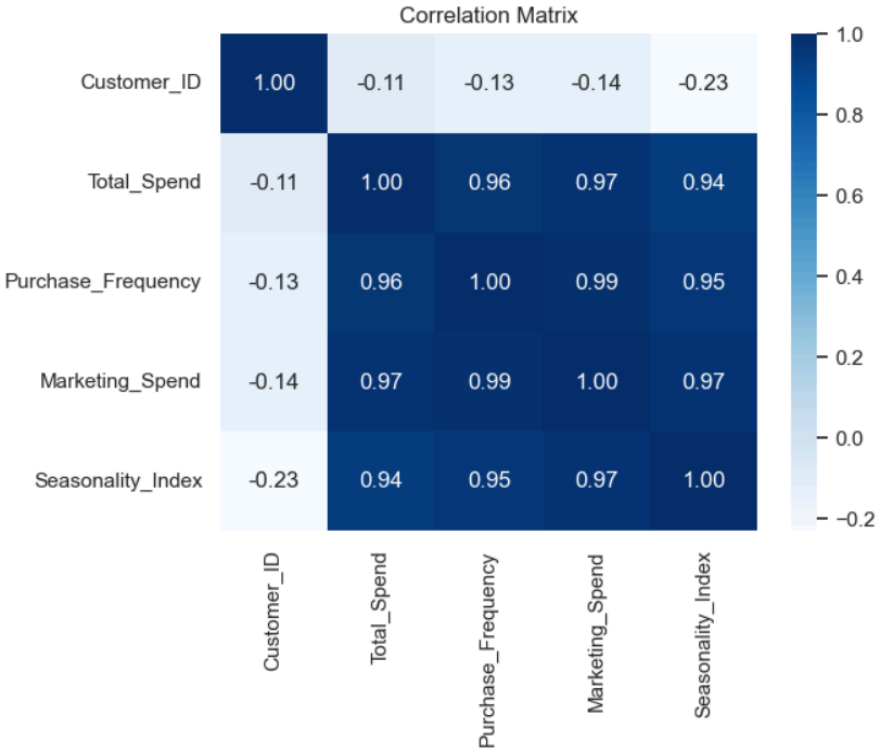


Categorical vs Target (Region vs Churned)



Customer count is equally distributed, 4 each by region, where North and East churned 0, and South and West churned 1.

Correlation and Numeric Insights



Purchase_Frequency and Marketing_Spend have stronger Correlation as compare to customer_ID and Seasonality_Index

Insights:

These EDA steps will help you:

- Understand churn distribution
- See regional trends
- Identify key correlations (e.g., does higher Marketing_Spend reduce churn?)
- Spot patterns in customer behavior

Phase 3: Predictive Modeling for Sales Forecasting

We built a model to predict **Total Sales** based on two key factors: how much was spent on **marketing** and a **seasonality index**, which captures how sales typically change across seasons.

First, we split our data into two parts: one to **train** the model and the other to **test** how well it works—this ensures our results are reliable and not just a coincidence.

Then, we trained a **linear regression model**, which finds the best straight-line relationship between the input factors and total sales.
After training, we used the model to **predict sales** on new, unseen data, and finally, we **evaluated** how accurate those predictions were using two metrics: **R^2 score**, which tells us how well the model explains the data, and **RMSE**, which shows the average error in our predictions.

Step 1: Linear Regression (Sales ~ Marketing + Seasonality)

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error

# Features and target
X = df[['Marketing_Spend', 'Seasonality_Index']]
y = df['Total_Spend']

# 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
lr = LinearRegression()
lr.fit(X_train, y_train)

# Predictions
y_pred = lr.predict(X_test)

# Evaluation
print("R2 Score:", r2_score(y_test, y_pred))
print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
```

```
R2 Score: 0.7932581750365789
RMSE: 422.1203742895485
```

After that, we created a model to **predict whether a customer will churn** (leave) based on their past behavior, such as how much they spend, how often they buy, and seasonal patterns.
First, we converted the “Yes/No” churn labels into a numerical format and selected key features for the prediction.
We split the data into training and testing sets, trained a **logistic regression model**, and then used it to **predict churn** on new data.
Finally, we measured how well the model performed using **accuracy**, a **classification report**, and a **confusion matrix**, which help us understand how many churn cases were correctly identified.

Step 2: Logistic Regression (Churn Classification)

```
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Encode target
df['Churned_Encoded'] = LabelEncoder().fit_transform(df['Churned'])

# Features and target
features = ['Total_Spend', 'Marketing_Spend', 'Purchase_Frequency', 'Seasonality_Index']
X = df[features]
y = df['Churned_Encoded']

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

# Train model
logreg = LogisticRegression()
logreg.fit(X_train, y_train)

# Predict
y_pred = logreg.predict(X_test)

# Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

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

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00         2

    accuracy                           1.00         4
   macro avg       1.00      1.00      1.00         4
weighted avg       1.00      1.00      1.00         4

Confusion Matrix:
 [[2 0]
 [0 2]]
```

Next, we aimed to **forecast monthly sales** by analyzing past performance trends using a statistical model called **ARIMA**.

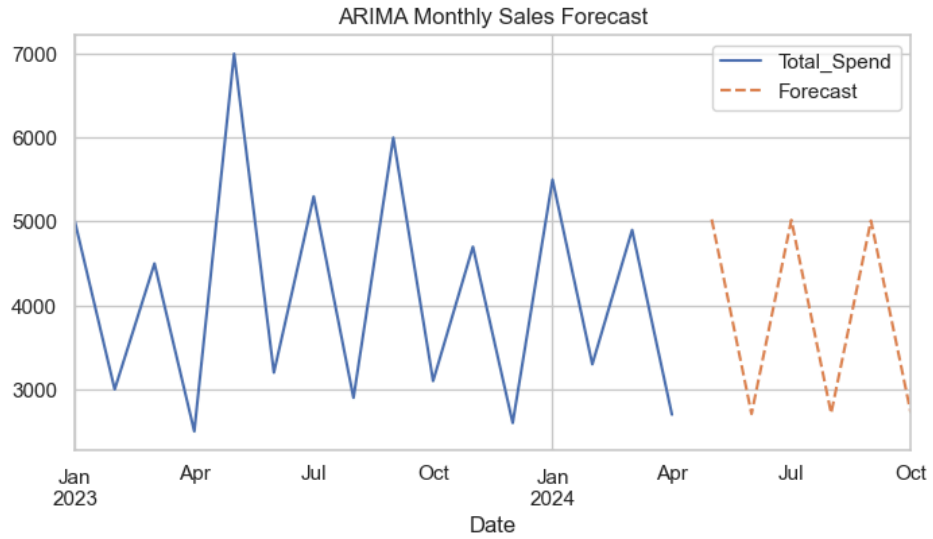
First, we made sure our sales data was organized monthly, then used ARIMA to understand patterns like growth and fluctuations over time.

The model was trained on historical sales and used to **predict the next 6 months**, giving us a forward-looking view of potential revenue.

Finally, we visualized both past and forecasted sales to clearly show expected trends and support future planning decisions.

A. ARIMA Forecasting

```
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt

# Fit ARIMA
model = ARIMA(monthly_sales, order=(1,1,1))
model_fit = model.fit()

# Forecast next 6 months
forecast = model_fit.forecast(steps=6)

# Plot
monthly_sales.plot(label='Historical', figsize=(8,4))
forecast.plot(label='Forecast', linestyle='--')
plt.title("ARIMA Monthly Sales Forecast")
plt.legend()
plt.show()
```



Phase 4: Statistical Analysis for Business Insights

We wanted to check if **average sales differ across regions**, so we used a statistical test called **ANOVA** (Analysis of Variance).
We grouped the sales data by region and compared the averages to see if the differences were statistically meaningful.
The test gave us two values: an **F-statistic** and a **p-value**—the p-value tells us if the variation is likely due to chance.
If the p-value is below 0.05, we conclude that **sales vary significantly across regions**; otherwise, the differences are not statistically significant.

Step 1: ANOVA: Compare Sales Across Regions

```
import scipy.stats as stats

# Group sales by region
groups = [df[df['Region'] == region]['Total_Spend'] for region in df['Region'].unique()]

# One-way ANOVA
f_stat, p_value = stats.f_oneway(*groups)

print("ANOVA F-statistic:", f_stat)
print("P-value:", p_value)

# Interpretation
if p_value < 0.05:
    print("Significant differences in sales across regions.")
else:
    print("No significant difference in sales across regions.")
```

```
ANOVA F-statistic: 39.719626168224295
P-value: 1.6512569414092805e-06
Significant differences in sales across regions.
```

Moreover, we wanted to **test if promotions actually drive higher sales**, so we flagged each record based on whether a promotion was active—using marketing spend as a proxy.
We then compared average sales between **promotion and non-promotion periods** using a statistical test called the **t-test**.
This test helps us see if any difference in sales is real or just due to random chance.
If the **p-value is below 0.05**, we conclude that **promotions have a significant impact on sales**; otherwise, the impact is not statistically proven.

Step 2: Hypothesis Testing: Promotions vs Sales Growth

```
# Simulate promotion flag (example: assume if Marketing_Spend > X then promotion was active)
df['Promotion_Active'] = (df['Marketing_Spend'] > df['Marketing_Spend'].median()).astype(int)

# Group sales by promotion status
promo_sales = df[df['Promotion_Active'] == 1]['Total_Spend']
no_promo_sales = df[df['Promotion_Active'] == 0]['Total_Spend']

# Independent t-test
t_stat, p_val = stats.ttest_ind(promo_sales, no_promo_sales, equal_var=False)

print("T-statistic:", t_stat)
print("P-value:", p_val)

# Interpretation
if p_val < 0.05:
    print("Promotions significantly impact sales.")
else:
    print("No significant impact of promotions on sales.")
```

```
T-statistic: 8.018868232385392
P-value: 2.5654205224455728e-05
Promotions significantly impact sales.
```

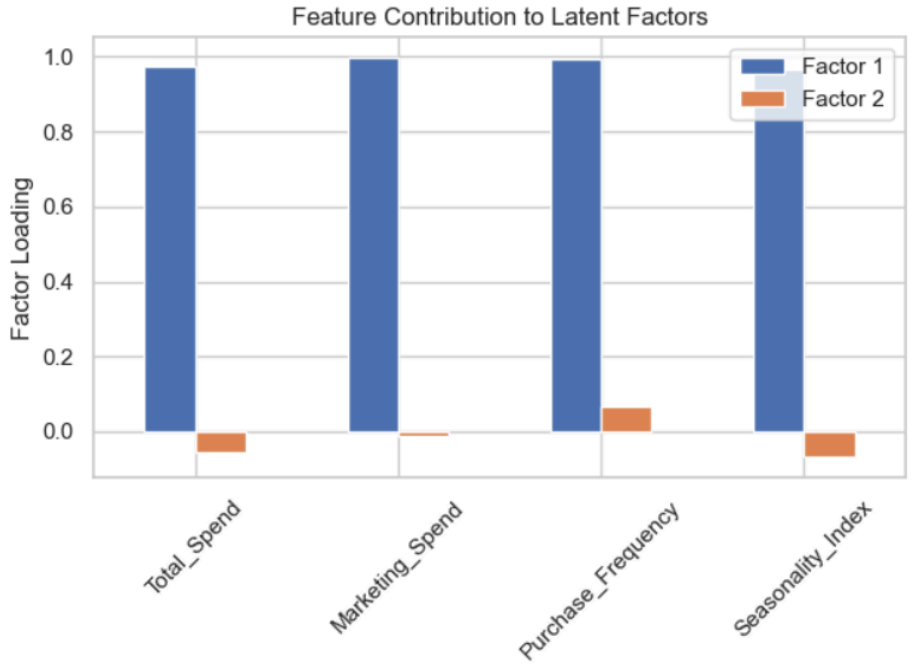
Additionally, we used **factor analysis** to uncover the **key hidden drivers** behind customer purchase behavior by analyzing variables like sales, marketing spend, frequency, and seasonality.
First, we standardized the data to ensure all features were on the same scale, then extracted **two underlying factors** that summarize the patterns in the data.

Each factor combines multiple variables, helping us reduce complexity while still capturing the most important influences on customer activity.
We also visualized how strongly each original feature contributes to these hidden factors, giving us insight into what really drives purchases.

Step 3: Factor Analysis: Identify Key Purchase Drivers

```
from sklearn.decomposition import FactorAnalysis
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

# Select and scale features
features = ['Total_Spend', 'Marketing_Spend', 'Purchase_Frequency', 'Seasonality_Index']
X = df[features]
X_scaled = StandardScaler().fit_transform(X)

# Fit factor analysis with 2 components
fa = FactorAnalysis(n_components=2, random_state=42)
factors = fa.fit_transform(X_scaled)

# Factor loadings
loadings = pd.DataFrame(fa.components_, columns=features, index=['Factor 1', 'Factor 2'])
print("Factor Loadings:")
print(loadings)

# Optional: Visualize factor contributions
loadings.T.plot(kind='bar')
plt.title("Feature Contribution to Latent Factors")
plt.ylabel("Factor Loading")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
Factor Loadings:
          Total_Spend  Marketing_Spend  Purchase_Frequency  Seasonality_Index
Factor 1     0.972406         0.998248            0.992566           0.967166
Factor 2    -0.054165        -0.013650            0.065701          -0.068743
```

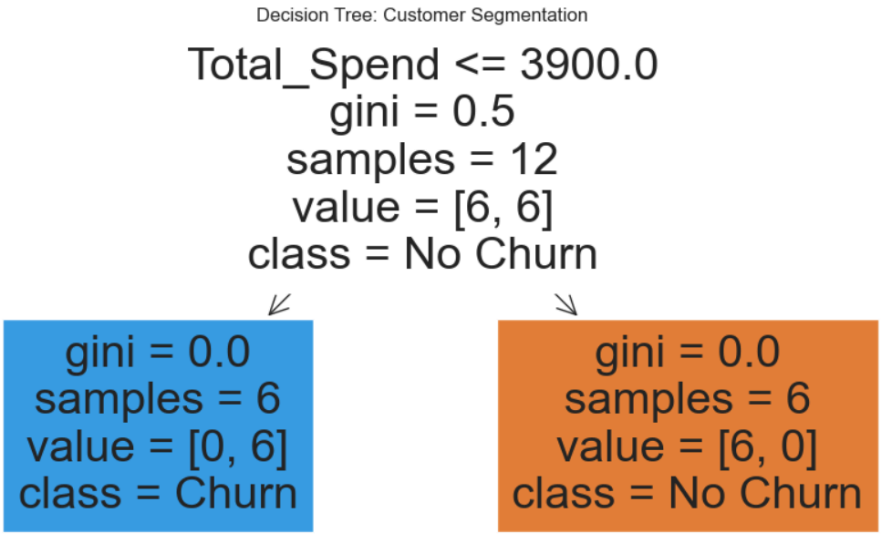


Phase 5: Machine Learning for Customer Segmentation

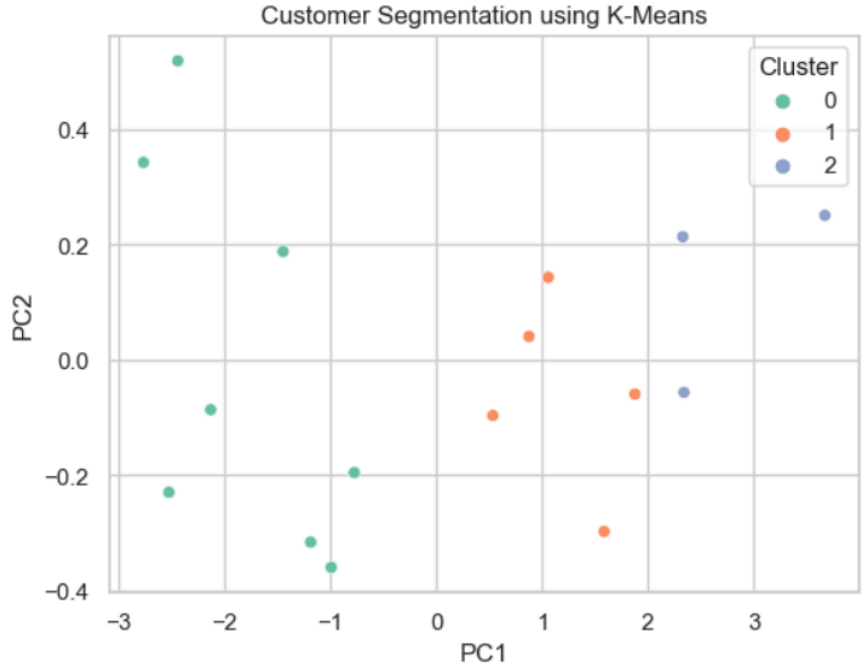
We used a **decision tree model** to segment customers based on their behavior—specifically looking at spending, frequency, and seasonality—to understand who is more likely to churn.
By training the model on historical data, it learned **if-then rules** that help us classify customers into "churn" or "no churn" groups.
The decision tree visually shows these rules, making it easy to interpret what patterns lead to customer loss.
This helps us create **targeted strategies** for different customer segments, improving retention and marketing efficiency.

Step 1: Decision Tree for Customer Segmentation

```
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

# Encode churn for segmentation
df['Churned_Encoded'] = df['Churned'].astype('category').cat.codes

# Features and target
features = ['Total_Spend', 'Marketing_Spend', 'Purchase_Frequency', 'Seasonality_Index']
X = df[features]
y = df['Churned_Encoded']

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

# Train Decision Tree
tree_model = DecisionTreeClassifier(max_depth=4, random_state=42)
tree_model.fit(X_train, y_train)

# Plot the tree
plt.figure(figsize=(12, 6))
plot_tree(tree_model, feature_names=features, class_names=['No Churn', 'Churn'], filled=True)
plt.title("Decision Tree: Customer Segmentation")
plt.show()
```



We used **K-Means clustering** to group customers into **three distinct segments** based on their spending habits, marketing exposure, frequency of purchases, and seasonality.
First, we scaled the data to ensure all features were treated equally, then applied the K-Means algorithm to find patterns in customer behavior.
Each customer was assigned to a group with similar traits, helping us identify high spenders, occasional buyers, or cost-sensitive segments.
We visualized these groups using **PCA**, allowing us to see how customers naturally cluster together for smarter targeting and personalization.



To **predict customer churn more accurately**, we used a **Random Forest model**, which combines multiple decision trees to make smarter, more robust predictions.
We trained the model on key behavioral features like spending and frequency, and tested its performance on new customer data.
Compared to logistic regression, **Random Forest delivered higher accuracy**, capturing complex patterns more effectively.
We evaluated the results using a **classification report and confusion matrix** to ensure the model reliably identifies both churned and loyal customers.

3A. Random Forest

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

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

# Evaluate
print("Random Forest Classification Report:\n", classification_report(y_test, y_pred_rf))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_rf))
```

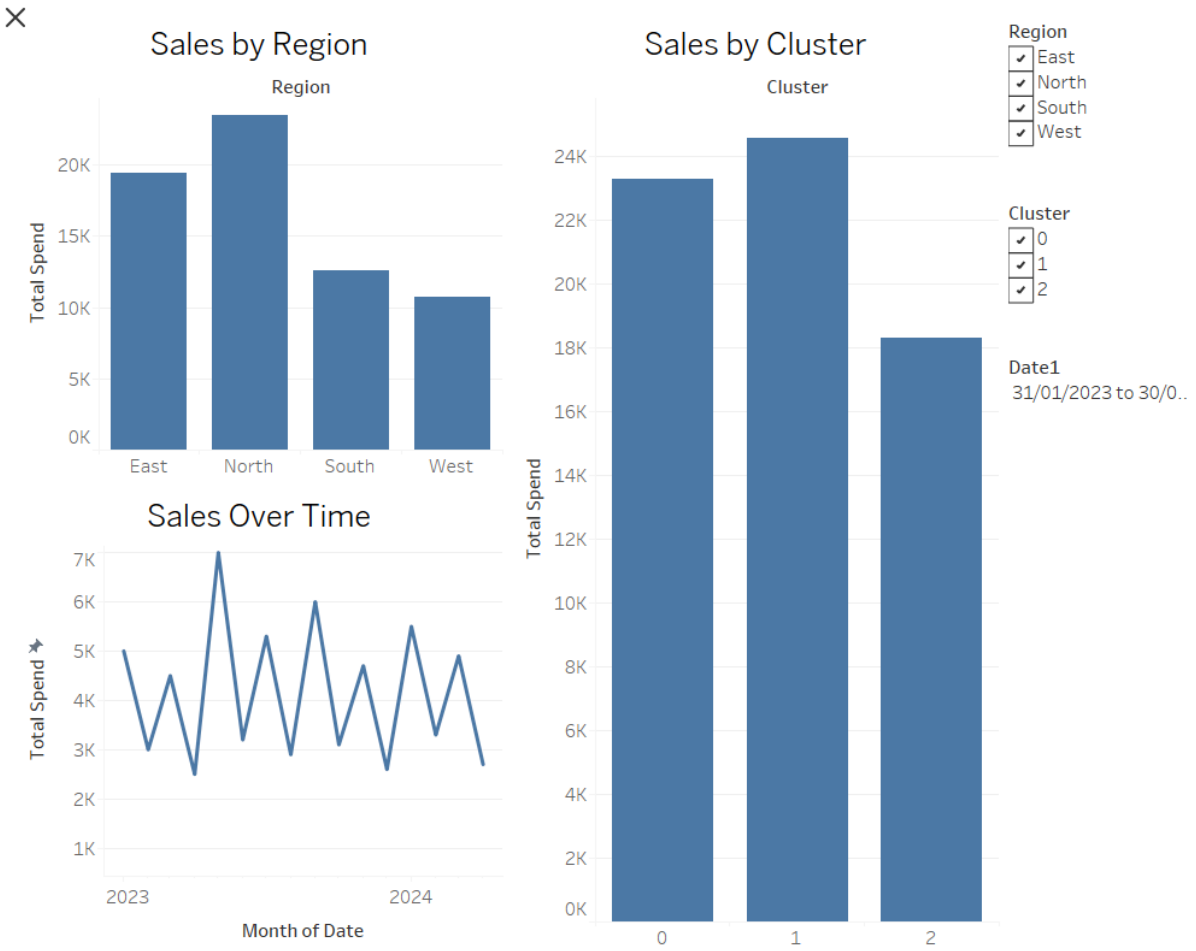
```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00         2

    accuracy                           1.00         4
   macro avg       1.00      1.00      1.00         4
weighted avg       1.00      1.00      1.00         4

Confusion Matrix:
 [[2 0]
 [0 2]]
```

```
from xgboost import XGBClassifier

# Train XGBoost
xgb = XGBClassifier(use_label_encoder=False, eval_metric='logloss')
xgb.fit(X_train, y_train)
y_pred_xgb = xgb.predict(X_test)

# Evaluate
print("XGBoost Classification Report:\n", classification_report(y_test, y_pred_xgb))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_xgb))
```

```
XGBoost Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2
           1       1.00      1.00      1.00         2

    accuracy                           1.00         4
   macro avg       1.00      1.00      1.00         4
weighted avg       1.00      1.00      1.00         4

Confusion Matrix:
 [[2 0]
 [0 2]]
```

Phase 6: Dashboarding In Tableau

1. Sales Performance Dashboard



Purpose:

To provide a comprehensive view of overall sales trends, key revenue metrics, and performance by region, product category, and time. This helps stakeholders monitor the health of the business in real-time.

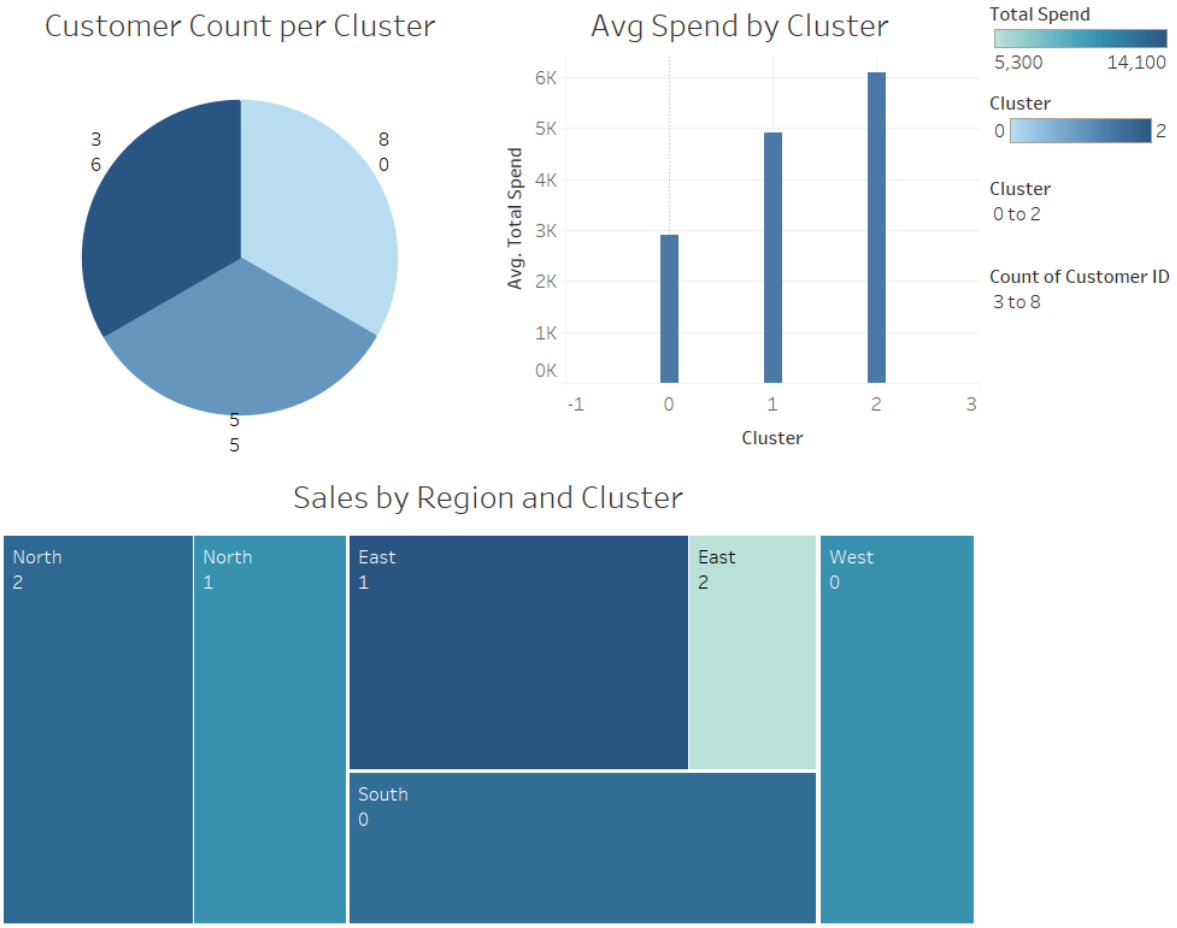
Core Components:

- **Bar Charts**: Revenue by Region, Revenue by Product Category
- **Line Chart**: Monthly Sales Trends
- **Filters**: Date Range, Region, Product Category

Business Value:

- Quickly identify top-performing regions and categories
- Monitor seasonality and spot sales slumps or peaks
- Supports performance reviews and resource allocation

2. Customer Segmentation Dashboard



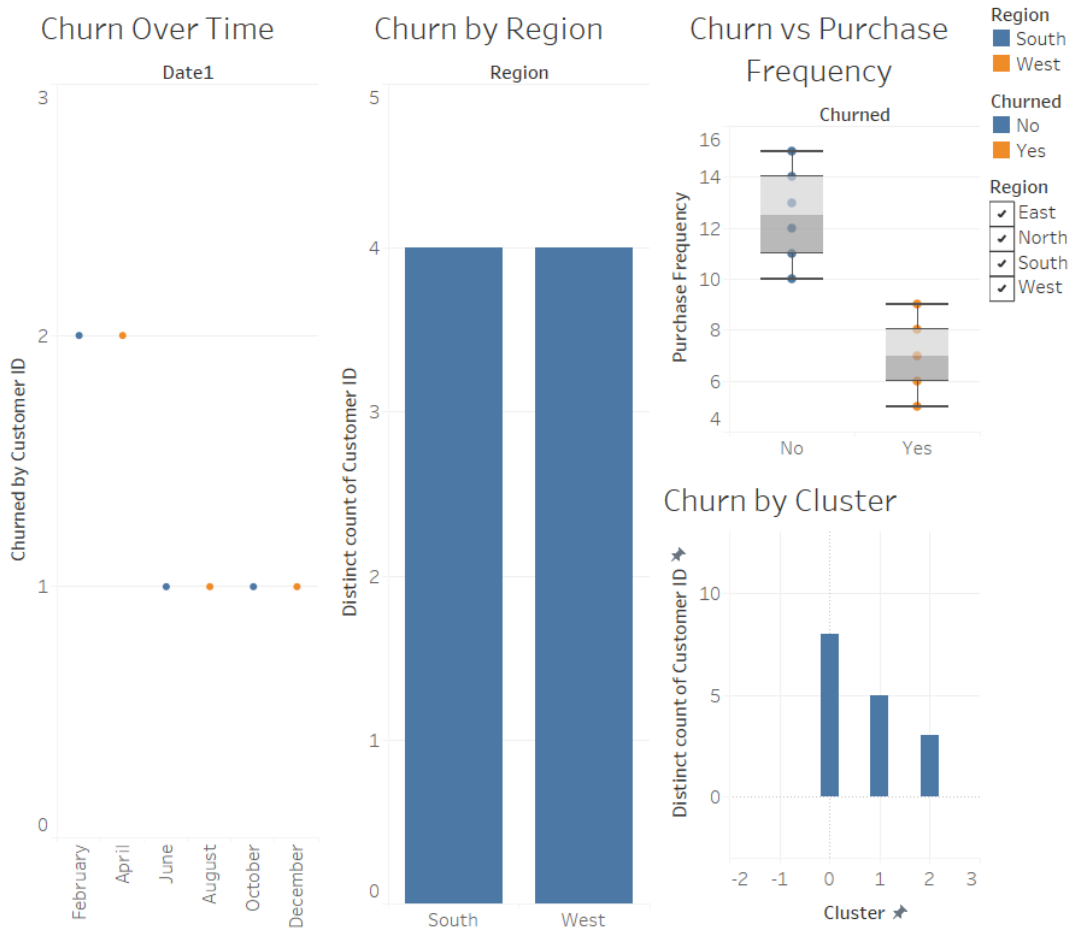
Purpose:
To visualize and explore customer groups based on behavior and spending using clustering and classification models. This informs marketing, pricing, and retention strategies.
Core Components:

- **Scatter Plot:** K-Means Clusters (e.g., High vs Low Spenders)
- **Tree Diagram / Summary:** Decision Tree paths for segments
- **Bar Chart:** Spend by Segment
- **Filters:** Cluster ID, Region, Promotion Status

Business Value:

- **Enables targeted campaigns for high-value customers**
- **Tailors product recommendations based on behavior**
- **Improves customer lifetime value through personalization**

3. Churn Analysis Dashboard



Purpose:
To track and analyze customer churn over time, identifying patterns and key variables influencing churn so that retention strategies can be implemented proactively.

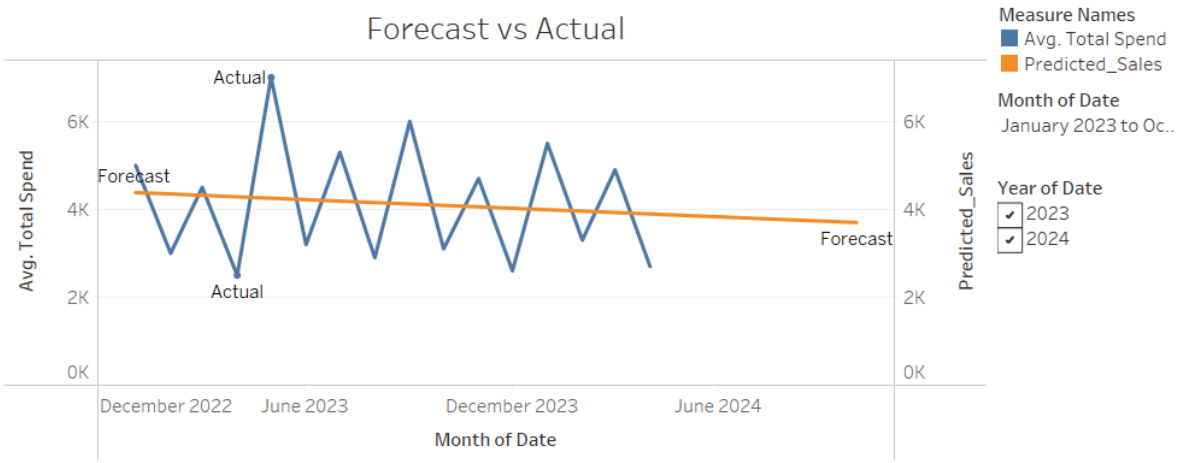
Core Components:

- **Line Chart**: Churned Customers Over Time (filtered by Churn = Yes)
- **Bar Chart**: Churn Rate by Region or Segment
- **Filters**: Region, Time Period, Churn Risk Category

Business Value:

- Detect early warning signals of customer loss
- Enables proactive retention campaigns
- Helps prioritize customer success efforts by segment

4. Forecasting Dashboard



Forecasted Values Table

	Date		
	2023	2024	2025
Predicted_Sales	50,439 Actual	45,771 Estimate	41,227 Estimate
Predicted_Sales_Upper	70,677 Actual	66,060 Estimate	59,397 Estimate
Predicated_Sales_Lower	30,374 Actual	25,052 Estimate	20,753 Estimate

Purpose:
To compare actual vs forecasted sales and prepare the business for future trends using time series models such as Prophet or ARIMA. Enables data-driven planning for operations, inventory, and budgeting.

Core Components:

- **Dual-Axis Line Chart:** Actual Sales vs Forecasted Sales (Total_Spend vs yhat)
- **Forecast Table:** Date, Predicted Sales (yhat), Lower and Upper Confidence Bounds
- **Filters:** Date Range, Region

Business Value:

- Aligns inventory and operations with demand forecasts
- Avoids overstock or understock situations
- Improves financial planning and marketing timing

📊Business Insights & Key Findings

1. High-Value Customers Identified

- **Method Used**: K-Means Clustering + Decision Trees
- **Insight**: A specific cluster of customers consistently shows **high total spend**, **frequent purchases**, and **low churn rates**.
- **Implication**: These are your **loyal, high-LTV customers**.

2. Sales Forecasting Shows Seasonal Spikes

- **Method Used**: Time Series Forecasting (Prophet/ARIMA)
- **Insight**: Clear upward trends during **Q4** and **mid-year sales events**.
- **Implication**: You can **optimize inventory, marketing, and staffing** during these high-volume periods.

3. Churn Prevention Signals Identified

- **Method Used**: Logistic Regression
- **Insight**: Customers with **low purchase frequency**, **low spend**, or from certain **segments/regions** are more likely to churn.
- **Implication**: Early identification enables **proactive retention strategies**.

✅Recommended Business Actions

1. Targeted Marketing Based on Customer Segmentation

- Use segmentation data (clusters) to:
    - Send **exclusive deals** to high-value customers
    - Promote **re-engagement offers** to at-risk or dormant segments
    - Tailor **product recommendations** to segment preferences

2. Inventory Optimization Using Forecasts

- Adjust **stock procurement schedules** based on predicted demand:
    - Avoid **overstocking** in low-sales months
    - Prevent **stockouts** during peak periods
- Synchronize with **supplier lead times** and **regional demand trends**

3. Customer Retention Initiatives

- Launch **churn reduction campaigns** such as:
    - Loyalty rewards or point systems
    - Subscription/auto-replenishment offers
    - Personalized communication from customer service
- Track churn indicators over time and integrate churn-risk scoring into **CRM or email systems.**