Customer Segmentation and Price Optimization Project

# 1. Introduction

In this project, we aim to use data-driven techniques to segment customers and optimize pricing strategies.   
Customer segmentation helps identify distinct groups within a customer base, while price optimization identifies the best pricing strategy to maximize sales and profit.   
The two goals are interrelated because different customer segments may respond differently to price changes. By combining both segmentation and pricing, businesses can   
tailor their pricing strategy for different groups, thereby maximizing profitability.

# 2. Project Objectives

The project has two key objectives:  
1. Customer Segmentation: Identify distinct groups of customers based on their purchasing behavior, demographics, and other relevant factors.  
2. Price Optimization: Predict the sales volume for various pricing and discount scenarios and determine the optimal pricing strategy that maximizes profit.

# 3. Dataset

## 3.1 Data Overview

The dataset used for this project includes customer demographic and transactional features. The key features are:  
- CLV (Customer Lifetime Value): A measure of how much revenue a customer is expected to generate.  
- Average Order Value: The average dollar amount spent by a customer in a transaction.  
- Purchase Frequency: The number of purchases made by the customer over a certain period.  
- Churn Risk: The probability that a customer will stop buying from the business.  
- Price Sensitivity: A measure of how sensitive a customer is to changes in price.  
- Income: The customer's income level.  
- Age: The age of the customer.

## 3.2 Data Preprocessing

The data was preprocessed to handle missing values, normalize numerical features, and encode categorical variables. Some key transformations included:  
- Normalization of continuous variables like CLV, Income, and Average Order Value.  
- Encoding of categorical variables where necessary (e.g., categorical customer groups).

# 4. Methodology

## 4.1 Customer Segmentation

### 4.1.1 Feature Selection

For customer segmentation, we used the following features from the dataset:

features = df\_encoded[['CLV', 'Avg\_Order\_Value', 'Purchase\_Frequency',  
 'Transaction\_Frequency', 'Churn\_Risk', 'Price\_Sensitivity', 'Age', 'Income']]

### 4.1.2 Clustering

We applied the KMeans Clustering algorithm to segment customers into distinct groups. The number of clusters was chosen using the Elbow Method,   
which evaluates how the sum of squared distances (inertia) decreases as the number of clusters increases.

### 4.1.3 Visualization

We used Plotly to visualize the clusters, helping us to explore how the customers are grouped based on their key features.

## 4.2 Price Optimization

### 4.2.1 Price Sensitivity Model

In the price optimization phase, we developed a regression model to predict sales volume under different price and discount scenarios.   
The goal was to optimize profit by finding the combination of base price and discount that yields the maximum profit.

### 4.2.2 Features for Prediction

For the price optimization model, we used the following features:  
- Base Price  
- Discount  
- Final Price (calculated as Base Price \* (1 - Discount))  
- Competitor Pricing  
- Profit Margin  
- Transaction Frequency  
- Elasticity of Demand

### 4.2.3 Model Training

We used a Random Forest Regressor to predict sales volume for each pricing scenario.   
The model was trained using the historical data, and key metrics like Mean Absolute Error (MAE) and R-squared were used to evaluate model performance.

### 4.2.4 Price and Discount Optimization

Once the model was trained, we created a grid of price and discount combinations to predict the sales volume and calculate profit for each scenario.   
The goal was to identify the combination of base price and discount that maximizes profit.

# 5. Results

## 5.1 Customer Segmentation Insights

Based on the clustering algorithm, we identified five distinct customer segments. These segments were defined by attributes such as CLV, Price Sensitivity,   
and Churn Risk. By analyzing the clusters, we gained insights into which customer groups are more price-sensitive, have higher CLV, or are at risk of churn.

## 5.2 Optimal Pricing Strategy

After running the price optimization model, the results showed that the optimal pricing strategy was:  
- Optimal Base Price: $100.00  
- Optimal Discount: 10%  
- Maximum Profit: $246.48  
- Predicted Sales Volume: 12 units  
  
An example scenario was created where the company was offering a 10% discount on a base price of $100. Under this pricing strategy, the predicted sales volume was 12 units,   
and the total profit generated was $246.48. Other pricing combinations (e.g., 20% discount on a base price of $90) were tested but resulted in lower profitability.

# 6. Usage

## 6.1 Cloning the Repository

To run this project locally:  
```bash  
git clone https://github.com/your-repo/customer-segmentation-price-optimization.git  
cd customer-segmentation-price-optimization  
```

## 6.2 Installing Dependencies

Install the required Python packages by running:  
```bash  
pip install -r requirements.txt  
```

## 6.3 Running the Project

- Customer Segmentation: Use the provided script to train and visualize customer segments:  
```bash  
python customer\_segmentation.py  
```  
  
- Price Optimization: Run the pricing model to predict sales volumes and calculate optimal prices:  
```bash  
python price\_optimization.py  
```

# 7. Future Work

- Dynamic Pricing for Each Segment: Implement customer-specific pricing strategies based on segment characteristics.  
- Real-time Price Optimization: Incorporate real-time factors such as seasonality and competitor prices to adjust pricing dynamically.  
- Testing Other Models: Explore more advanced models like Gradient Boosting Machines (GBMs) for improved prediction accuracy.

# 8. Dependencies

The project requires the following Python libraries:  
- pandas  
- numpy  
- scikit-learn  
- plotly  
- matplotlib  
- seaborn  
  
Make sure to install the necessary dependencies using:  
```bash  
pip install -r requirements.txt  
```