Customer Segmentation Using Unsupervised Learning

# 1. Introduction

Machine Learning has revolutionized data analysis and decision-making across industries.   
Unsupervised learning, a subfield of machine learning, is used to identify hidden patterns or intrinsic structures in data without predefined labels.   
This project leverages unsupervised learning to uncover customer segments in the retail domain using the Mall Customer Segmentation dataset.

# 2. Objective

To segment customers into different groups based on their demographic and behavioral data using K-Means clustering algorithm, thereby enabling businesses to understand their customer base and tailor their marketing strategies accordingly.

# 3. Dataset

Name: Mall Customer Segmentation Dataset  
Source: Kaggle (https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial)  
Features:  
- CustomerID (ignored in analysis)  
- Gender  
- Age  
- Annual Income (k$)  
- Spending Score (1-100)

# 4. Methodology

1. Data Preprocessing: Encoding categorical features and standardizing numerical data.  
2. Determining optimal number of clusters using the Elbow Method.  
3. Applying K-Means Clustering for segmentation.  
4. Visualizing clusters using PCA (Principal Component Analysis).

# 5. Algorithm Used

K-Means Clustering:  
- An iterative, partition-based clustering algorithm that assigns each data point to one of K groups.  
- The optimal number of clusters is selected using the Elbow Method.  
- PCA (Principal Component Analysis) is used for 2D visualization of clusters.

# 6. Results

- Successfully identified 5 distinct customer segments.  
- Plotted customer clusters using PCA to visualize separation and grouping.  
- Grouped customers based on similarities in spending and income behaviors.

# 7. Applications

- Personalized Marketing  
- Customer Relationship Management  
- Improved Inventory Planning  
- Enhanced Customer Experience

# 8. Conclusion

This project demonstrates the capability of unsupervised learning in discovering meaningful patterns in customer behavior.   
K-Means clustering successfully segmented customers into distinct groups, offering valuable insights for business strategies.

# 9. Future Work

- Explore alternative clustering algorithms such as DBSCAN and Hierarchical Clustering.  
- Incorporate additional behavioral features like product categories, time spent, and online activity.  
- Build an interactive dashboard for business teams to explore segmentation results.

# 10. References

- Kaggle Dataset: https://www.kaggle.com/datasets/vjchoudhary7/customer-segmentation-tutorial  
- Scikit-learn Documentation: https://scikit-learn.org/stable/  
- Research Papers on Unsupervised Learning

# 11. Python Implementation (Code)

## Importing Libraries and Loading Data

import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
from sklearn.preprocessing import StandardScaler  
from sklearn.cluster import KMeans  
from sklearn.decomposition import PCA  
  
# Load dataset  
url = "https://raw.githubusercontent.com/shubham0204/Dataset\_Repository/main/mall\_customers.csv"  
df = pd.read\_csv(url)

## Data Preprocessing

# Preprocess data  
df.drop('CustomerID', axis=1, inplace=True)  
df['Gender'] = df['Gender'].map({'Male': 0, 'Female': 1}) # Encode Gender  
features = ['Gender', 'Age', 'Annual Income (k$)', 'Spending Score (1-100)']  
X = df[features]  
  
# Standardize data  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)

## Finding Optimal Number of Clusters (Elbow Method)

# Elbow Method to find optimal k  
wcss = []  
for i in range(1, 11):  
 kmeans = KMeans(n\_clusters=i, init='k-means++', random\_state=42)  
 kmeans.fit(X\_scaled)  
 wcss.append(kmeans.inertia\_)  
  
plt.plot(range(1, 11), wcss, marker='o')  
plt.title('Elbow Method')  
plt.xlabel('Number of Clusters')  
plt.ylabel('WCSS')  
plt.show()

## Applying KMeans Clustering

# Apply KMeans with optimal k  
k = 5  
kmeans = KMeans(n\_clusters=k, init='k-means++', random\_state=42)  
clusters = kmeans.fit\_predict(X\_scaled)  
df['Cluster'] = clusters

## Dimensionality Reduction and Visualization

# Optional: PCA for 2D visualization  
pca = PCA(n\_components=2)  
reduced = pca.fit\_transform(X\_scaled)  
df['PCA1'] = reduced[:, 0]  
df['PCA2'] = reduced[:, 1]  
  
# Plot clusters  
plt.figure(figsize=(8, 6))  
sns.scatterplot(data=df, x='PCA1', y='PCA2', hue='Cluster', palette='Set2')  
plt.title('Customer Segments (PCA Visualization)')  
plt.show()