## **Data Science for Product Sales Prediction & Clustering**

### 

### **Introduction:**

In today’s competitive market, businesses must analyze sales trends and optimize product inventory to maximize revenue. This project applies machine learning techniques to predict sales and identify product groups using supervised and unsupervised learning.

**The project focuses on:**

1) Supervised Learning: Using Random Forest Regression to predict monthly sales based on product features (Price, Stock Available, and Category).

2) Unsupervised Learning: Using K-Means Clustering to group products based on their pricing, availability, and sales performance.

By leveraging data science, businesses can forecast sales, improve stock management, and optimize pricing strategies.

### **Dataset Overview:**

* Rows: 800
* Columns: 6
* Features:  
  + ProductID (int) – Unique identifier for each product
  + Product (object) – Product name (not needed for modeling)
  + Category (object) – Product category (e.g., Grocery, Clothing, Toys)
  + Price (int) – Price of the product
  + Stock Available (int) – Quantity available in stock
  + Sales (Last Month) (int) – Sales count for the last month (Target for regression)

## **Objective:**

The project applies supervised learning (Random Forest Regression) to predict product sales and unsupervised learning (K-Means Clustering) to group similar products based on their features.

## **Data Preprocessing**

* Loaded dataset larger\_product\_sales.csv.
* Encoded categorical variables (Category) into numerical format.
* Standardized numerical features (Price, Stock Available, Category) for better performance.
* Split data into training (80%) and testing (20%) sets for regression.

## **Supervised Learning:** Sales Prediction (Random Forest Regression)

### **Goal:**

Train a model to predict sales (Sales (Last Month)) based on Price, Stock Available, and Category.

### Steps:

1. Used Random Forest Regression (100 decision trees).
2. Trained the model on historical data.
3. Evaluated performance using:  
   * Mean Absolute Error (MAE) – Measures average prediction error.
   * Mean Squared Error (MSE) – Penalizes large errors more.

### **Results:**

* The model provides better predictions than basic linear models but can be improved by adding more features (e.g., promotions, customer reviews).

## **3) Unsupervised Learning:** Product Clustering (K-Means)

### **Goal:**

Identify groups of similar products based on Price, Stock, and Sales trends.

### **Steps:**

1. Used K-Means Clustering with k=3 (3 groups).
2. Assigned each product to a cluster based on its sales behavior.
3. Visualized clusters using a scatter plot (Price vs. Sales, colored by cluster).

### **Insights:**

* Cluster 1: High-price, high-sales products.
* Cluster 2: Mid-range products with moderate sales.
* Cluster 3: Low-price, low-sales products.

## **4) Key Business Applications**

* Predicting sales helps businesses adjust stock levels and pricing strategies.
* Clustering helps identify best-selling products and optimize marketing.
* Stock Management: Helps identify overstocked or underperforming products.

**Step 1: Importing Required Libraries**

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.ensemble import RandomForestRegressor

from sklearn.cluster import KMeans

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error

import matplotlib.pyplot as plt

import seaborn as sns

**Step 2: Loading the Dataset**

df = pd.read\_csv("larger\_product\_sales.csv")

**Step 3: Encoding Categorical Data (Category)**

le = LabelEncoder()

df['Category'] = le.fit\_transform(df['Category'])

**Step 4: Defining Features (X) and Target (y)**

X = df[['Price', 'Stock Available', 'Category']]

y = df['Sales (Last Month)']

**Step 5: Standardizing Features**

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

**Step 6: Splitting Data into Training and Testing Sets**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

**Step 7: Supervised Learning (Random Forest Regression)**

regressor = RandomForestRegressor(n\_estimators=100, random\_state=42)

regressor.fit(X\_train, y\_train)

y\_pred = regressor.predict(X\_test)

**Step 8: Evaluating Regression Model**

print("Mean Absolute Error:", mean\_absolute\_error(y\_test, y\_pred))

print("Mean Squared Error:", mean\_squared\_error(y\_test, y\_pred))

**Step 9: Unsupervised Learning (K-Means Clustering)**

kmeans = KMeans(n\_clusters=3, random\_state=42)

df['Cluster'] = kmeans.fit\_predict(X\_scaled)

**Step 10: Visualizing Clusters**

plt.figure(figsize=(8, 6))

sns.scatterplot(x=df['Price'], y=df['Sales (Last Month)'], hue=df['Cluster'], palette='viridis')

plt.title("K-Means Clustering of Product Sales")

plt.xlabel("Price")

plt.ylabel("Sales (Last Month)")

plt.show()

What This Plot Shows:

* X-Axis: Product Price
* Y-Axis: Sales (Last Month)
* Color: Different product clusters

## **Final Summary**

* Supervised Learning (Random Forest Regression):

1. Predicts Sales (Last Month) using Price, Stock, and Category.
2. Model performance is evaluated using MAE & MSE.

* Unsupervised Learning (K-Means Clustering):

1. Groups products into 3 clusters based on pricing and sales trends.
2. Visualization helps understand which products are similar in sales behavior.