Kings Coffee Intelligence System

This desktop application supports smart decision-making for a boutique coffee shop by applying data analytics to optimize operations. It utilizes customer and sales data to offer insights via four main analytics strategies—Descriptive, Diagnostic, Predictive, and Prescriptive—and helps staff visualize trends, forecast sales, and apply pricing strategies. This project focuses on optimizing a boutique coffee shop's operations through strategic data analysis. By applying four core types of analytics—descriptive, diagnostic, predictive, and prescriptive—the initiative identifies top-performing coffee types, uncovers patterns behind sales fluctuations, forecasts future demand based on monthly trends, and introduces dynamic pricing strategies to enhance revenue during off-peak hours. The approach uses customer and sales data to inform decisions around inventory management, staffing, promotional timing, and pricing adjustments. The goal is to create a smart, data-driven business model that not only meets customer preferences but also boosts profitability, reduces waste, and strengthens overall operational efficiency.

Descriptive - Daily coffee sales - Latte identified as the most popular beverage.

Diagnostic- Weekly sales drop pattern - Notable dips on midweek and Sundays for multiple items.

Predictive- Monthly sales forecasting - Latte and Cold Brew projected to maintain strong demand.

Prescriptive- Pricing strategy based on traffic trend - Dynamic pricing recommended during slow hours.

Problem Statement

A boutique coffee shop struggles with:

- **Inconsistent daily and weekly sales**
- **Lack of understanding of customer preferences**
- **Inefficient inventory and staffing decisions**
- **Revenue loss during off-peak hours**

Main Dataset Columns:

`Date`	**Transaction date**
`Coffee_Type`	Type of coffee sold (e.g., Latte, Cold Brew)
`Units_Sold`	Quantity sold each day
`Daily_Revenue`	Revenue generated per coffee type per day

Data Sources:

- Kaggle dataset or manually imported CSV files.

Output & Functionality

- GUI-based dashboard for:
    - Loading Kaggle or custom CSV data.
    - Running analysis at the click of a button.
    - Viewing clear visualizations (bar, line, forecast charts).
- Analysis modules include:
    - **Daily performance breakdowns**
    - **Trend detection across weeks**
    - **Sales forecasting per coffee type**
    - **Recommendations on pricing changes**

Purpose

To build a **data-driven operational model** that:

- Responds to customer trends.
- Minimizes waste and overstocking.
- Ensures optimal staffing schedules.
- Maximizes profitability—especially during slow sales windows.

Expected Outcomes

- Clear visibility into coffee performance per day/week/month.
- Anticipation of future demand patterns.
- Identification of low-revenue timings for pricing optimization.
- Empowered decision-making for promotions, staffing, and inventory.

Planning Phase

🎯 Goals

- Create a tool to support business decisions with data analytics.
- Improve sales, pricing, inventory, and staffing efficiency.
- Enable user-friendly access to insights via a GUI.

Scope

- Four types of analytics: Descriptive, Diagnostic, Predictive, Prescriptive.
- Data inputs: coffee sales, revenue, and daily logs.
- Outputs: Visual trends, sales forecasts, actionable suggestions.

Key Resources

- Tools: Python, Tkinter, Pandas, Matplotlib.
- Data: Coffee shop sales (CSV format, or Kaggle dataset).
- Audience: Coffee shop owners, managers, analysts.

Design Phase

GUI Layout

- Header: "Kings Coffee Intelligence System "
- Buttons:
  - Load Kaggle dataset
  - Import custom CSV
  - Descriptive, Diagnostic, Predictive, and Prescriptive analysis triggers
- Alerts: Message boxes for success, errors, or missing data
- Visualizations: Interactive plots via Matplotlib

Visual Design Considerations

- Use coffee-themed colours (sienna, crimson, mocha).
- Icons on buttons enhance interaction (e.g., 📊, 🔍).
- Responsive layout fits desktop-sized screen (500x430px).

Technology	Role
Tkinter -	GUI development
Pandas -	Data manipulation
Matplotlib -	Data visualization
CSV files -	Data source format

Module Breakdown

1. DataManager **Class**

- Loads predefined or custom datasets.
- Handles exceptions gracefully via message boxes.

2. CoffeeAnalytics **Class**

- descriptive(): Aggregates and visualizes coffee-type performance.
- diagnostic(): Tracks day-wise revenue trends.
- predictive(): Forecasts future revenue using rolling averages.
- prescriptive(): Highlights slow days for dynamic pricing.

3. CoffeeApp **Class**

- Constructs the main window interface.
- Connects GUI buttons to backend analysis functions.

Benefits:

- Boosted revenue through smarter pricing.
- Reduced waste and overstocking.
- Better customer satisfaction by stocking popular drinks.
- Enhanced decision-making with visual, actionable insights.

Outputs:

- Interactive charts showing revenue performance, trends, forecasts, and strategy insights.
- On-screen messages for guidance and error handling.

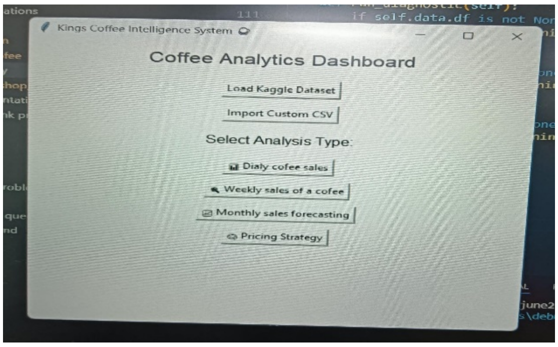
Code

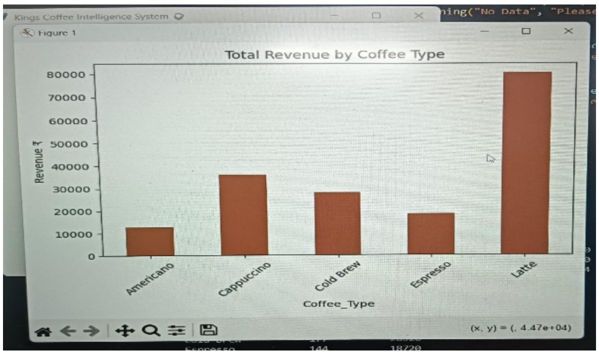
```
import tkinter as tk

from tkinter import filedialog, messagebox

import pandas as pd

import matplotlib.pyplot as plt


'''Database managment'''

class DataManager:

    def __init__(self):

        self.df = None


    def load_kaggle_data(self):

        try:

            self.df = pd.read_csv("coffee_shop_data.csv")  # Change to Kaggle file name

            messagebox.showinfo("Loaded", "Kaggle dataset loaded successfully.")

        except Exception as e:

            messagebox.showerror("Error", f"Could not load Kaggle data.\n{e}")


    def import_csv(self):

        path = filedialog.askopenfilename(filetypes=[("CSV files", "*.csv")])

        if path:

            try:

                self.df = pd.read_csv(path)

                messagebox.showinfo("Imported", "CSV file imported.")

            except Exception as e:

                messagebox.showerror("Error", f"CSV import failed.\n{e}")


'''Data Analysis'''

class CoffeeAnalytics:
```

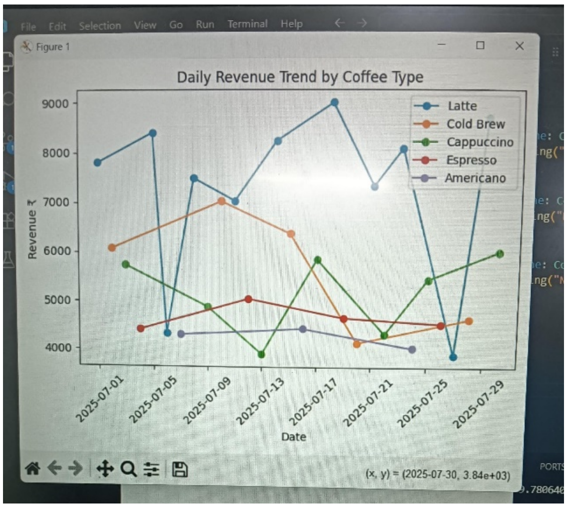
```
def __init__(self, df):
    self.df = df
    self.df['Date'] = pd.to_datetime(self.df['Date'], errors='coerce')

def descriptive(self): #Dialy Cofee Sales
    print("\n📊 Descriptive Analysis:")
    print(self.df.describe())
    print("\nBy Coffee Type:")
    print(self.df.groupby('Coffee_Type')[['Units_Sold', 'Daily_Revenue']].sum())
    self.df.groupby('Coffee_Type')['Daily_Revenue'].sum().plot(kind='bar', color='sienna')
    plt.title("Total Revenue by Coffee Type")
    plt.ylabel("Revenue ₹"); plt.xticks(rotation=45); plt.tight_layout(); plt.show()

def diagnostic(self): #Weekly Cofee Sales
    print("\n🔍 Diagnostic Analysis:")
    for c in self.df['Coffee_Type'].unique():
        sub = self.df[self.df['Coffee_Type'] == c]
        plt.plot(sub['Date'], sub['Daily_Revenue'], marker='o', label=c)
    plt.title("Daily Revenue Trend by Coffee Type")
    plt.xlabel("Date"); plt.ylabel("Revenue ₹")
    plt.legend(); plt.xticks(rotation=45); plt.tight_layout(); plt.show()

def predictive(self): #Monthly Sales forecasting
    print("\n📈 Predictive Analysis:")
    for c in self.df['Coffee_Type'].unique():
        sub = self.df[self.df['Coffee_Type'] == c].copy()
        sub['Forecast'] = sub['Daily_Revenue'].rolling(window=3).mean()
        plt.plot(sub['Date'], sub['Forecast'], label=f"{c} Forecast")
    plt.title("Forecasted Revenue (Moving Average)")
```

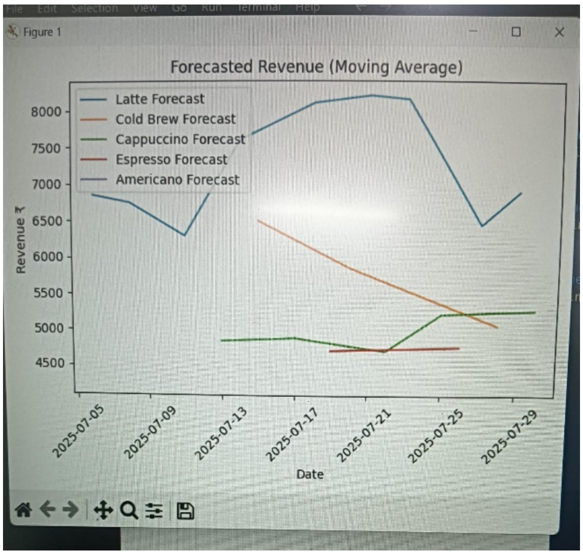
```
        plt.xlabel("Date"); plt.ylabel("Revenue ₹")
        plt.legend(); plt.xticks(rotation=45); plt.tight_layout(); plt.show()

    def prescriptive(self): #Pricing Strategy
        print("\n🧠 Prescriptive Analysis:")
        avg = self.df['Daily_Revenue'].mean()
        slow = self.df[self.df['Daily_Revenue'] < avg]
        print("Days Below Average:\n", slow[['Date', 'Coffee_Type', 'Daily_Revenue']])
        print("\nCoffee Types to Consider for Dynamic Pricing:")
        print(slow['Coffee_Type'].value_counts())
        slow.groupby('Coffee_Type')['Daily_Revenue'].mean().plot(kind='barh', color='crimson')
        plt.title("Target Types for Pricing Strategy")
        plt.xlabel("Avg Revenue ₹"); plt.tight_layout(); plt.show()

 #Graphical User Interface
class CoffeeApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Kings Coffee Intelligence System ☕ ")
        self.root.geometry("500x430")
        self.data = DataManager()

        tk.Label(root, text="Coffee Analytics Dashboard", font=("Helvetica", 16)).pack(pady=10)
        tk.Button(root, text="Load Kaggle Dataset", command=self.data.load_kaggle_data).pack(pady=5)
        tk.Button(root, text="Import Custom CSV", command=self.data.import_csv).pack(pady=5)
        tk.Label(root, text="Select Analysis Type:", font=("Helvetica", 12)).pack(pady=10)

        tk.Button(root, text="📊 Dialy cofee sales", command=self.run_descriptive).pack(pady=5)
        tk.Button(root, text="🔍 Weekly sales of a cofee", command=self.run_diagnostic).pack(pady=5)
        tk.Button(root, text="📈 Monthly sales forecasting",
command=self.run_predictive).pack(pady=5)

        tk.Button(root, text="🧠 Pricing Strategy", command=self.run_prescriptive).pack(pady=5)


    def run_descriptive(self):

        if self.data.df is not None: CoffeeAnalytics(self.data.df).descriptive()

        else: messagebox.showwarning("No Data", "Please load a dataset.")


    def run_diagnostic(self):

        if self.data.df is not None: CoffeeAnalytics(self.data.df).diagnostic()

        else: messagebox.showwarning("No Data", "Please load a dataset.")


    def run_predictive(self):

        if self.data.df is not None: CoffeeAnalytics(self.data.df).predictive()

        else: messagebox.showwarning("No Data", "Please load a dataset.")


    def run_prescriptive(self):

        if self.data.df is not None: CoffeeAnalytics(self.data.df).prescriptive()

        else: messagebox.showwarning("No Data", "Please load a dataset.")


#Main function

if __name__ == "__main__":

    root = tk.Tk()

    CoffeeApp(root)

    root.mainloop()
```

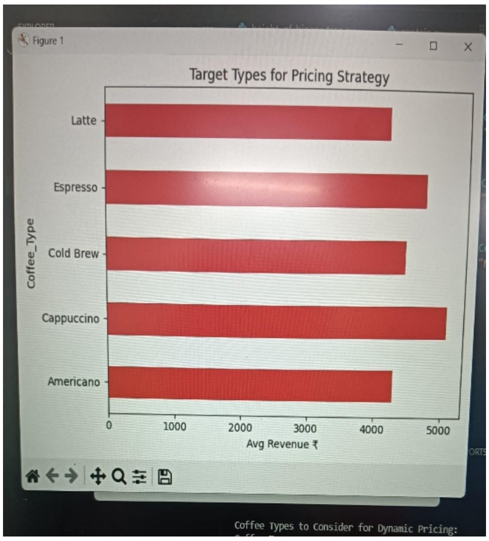
Output

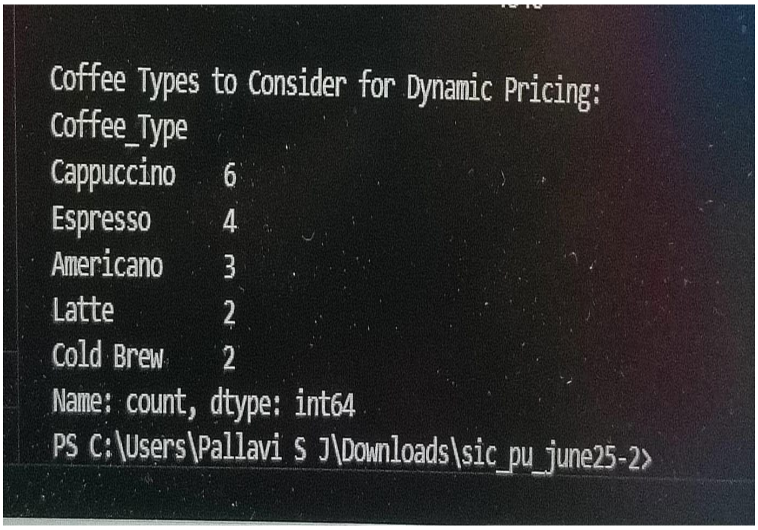












```
Coffee Types to Consider for Dynamic Pricing:
Coffee_Type
Cappuccino    6
Espresso      4
Americano     3
Latte         2
Cold Brew     2
Name: count, dtype: int64
PS C:\Users\Pallavi S J\Downloads\sic_pu_june25-2>
```

Closure

The *Kings Coffee Intelligence System* demonstrates the powerful intersection of data analytics and small business strategy. By applying descriptive, diagnostic, predictive, and prescriptive methods to real-world coffee shop data, it transforms raw figures into actionable insights. From identifying top-performing beverages and analyzing weekly sales patterns to forecasting future demand and recommending strategic pricing, this system empowers decision-makers to optimize inventory, reduce waste, improve customer satisfaction, and boost profitability. Its intuitive Python-based GUI ensures accessibility for non-technical users, making advanced analysis tools usable in everyday operations. Ultimately, this project offers a scalable blueprint for any boutique retail establishment seeking to become more data-savvy**.**

Bibliography

Here's a list of references and inspirations behind the project's concept and technology:

- Kaggle Coffee Shop Datasets *Used for testing and simulating realistic sales data.* https://www.kaggle.com
- Wes McKinney. *Python for Data Analysis. Key resource for understanding Pandas and data manipulation techniques.* ISBN: 978-1491957660
- Matplotlib Documentation *Used for visualizing analytics through charts and plots.* https://matplotlib.org/stable/index.html
- Tkinter GUI Programming *Essential for building the interactive desktop interface.* https://docs.python.org/3/library/tkinter.html
- Business Analytics Concepts *Inspired the analytical framework: descriptive, diagnostic, predictive, and prescriptive.* James R. Evans. *Business Analytics: Methods*