**DataMart Annual Operations Analysis**

**Introduction**

DataMart, a leading retail chain specializing in consumer electronics, seeks to analyze its branches' performance to drive better decision-making. This analysis focuses on revenue trends, employee satisfaction, and expenses, highlighting areas for improvement and success. Python's powerful analytical tools and visualizations are leveraged to derive actionable insights.

**Objectives**

1. Classify branches into performance categories (Excellent, Good, Needs Improvement) based on revenue metrics.
2. Analyze employee satisfaction data and its correlation with branch profitability.
3. Compute financial metrics such as total revenue, net income, and profit margins.
4. Identify revenue trends and anomalies for better forecasting.
5. Provide clear visualizations and a consolidated dashboard for quick decision-making.
6. Generate actionable recommendations for optimizing operations.

**Tools and Technologies**

* **Pandas**: For data manipulation, cleaning, and preprocessing.
* **NumPy**: For efficient numerical computations and array operations.
* **Matplotlib**: For creating static and detailed data visualizations.
* **Seaborn**: For enhanced statistical data visualization.

**Implementation**

# Importing Required Libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Data Initialization

branches = ["Branch A", "Branch B", "Branch C", "Branch D"]

monthly\_revenue = np.random.randint(30000, 150000, size=(12, len(branches)))

monthly\_expenses = np.random.randint(20000, 100000, size=(12, len(branches)))

employee\_satisfaction = np.random.uniform(3.0, 5.0, size=len(branches))

# Key Metrics

annual\_revenue\_per\_branch = monthly\_revenue.sum(axis=0)

total\_annual\_revenue = monthly\_revenue.sum()

avg\_employee\_satisfaction = employee\_satisfaction.mean()

# Branch Categorization

def categorize\_branch(revenue):

if revenue > 1\_000\_000:

return "Excellent"

elif 500\_000 <= revenue <= 1\_000\_000:

return "Good"

else:

return "Needs Improvement"

branch\_categories = [categorize\_branch(rev) for rev in annual\_revenue\_per\_branch]

branch\_data = pd.DataFrame({

"Branch": branches,

"Annual Revenue": annual\_revenue\_per\_branch,

"Category": branch\_categories

})

# Net Income Calculation

def calculate\_net\_income(revenue, expenses):

return revenue - expenses

net\_income\_per\_month = calculate\_net\_income(monthly\_revenue, monthly\_expenses)

# Revenue Data Extraction

second\_quarter\_revenue = monthly\_revenue[3:6, :]

last\_three\_months\_revenue = monthly\_revenue[-3:, :]

avg\_monthly\_revenue = monthly\_revenue.mean(axis=1)

high\_revenue\_months = monthly\_revenue > 100\_000

projected\_revenue\_next\_year = monthly\_revenue \* 1.1

combined\_revenue = np.concatenate((monthly\_revenue, projected\_revenue\_next\_year), axis=0)

# Visualizations

# 1. Monthly Revenue Trend of Top-Performing Branch

plt.figure(figsize=(10, 6))

plt.plot(monthly\_revenue[:, np.argmax(annual\_revenue\_per\_branch)], marker='o')

plt.title("Monthly Revenue Trend of Top-Performing Branch")

plt.xlabel("Month")

plt.ylabel("Revenue ($)")

plt.savefig("line\_plot\_top\_branch.png")

plt.show()

# 2. Total Annual Revenue by Branch

plt.figure(figsize=(10, 6))

plt.bar(branches, annual\_revenue\_per\_branch, color='skyblue')

plt.title("Total Annual Revenue by Branch")

plt.xlabel("Branches")

plt.ylabel("Revenue ($)")

plt.savefig("bar\_chart\_annual\_revenue.png")

plt.show()

# 3. Revenue Contribution by Branch

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

plt.pie(annual\_revenue\_per\_branch, labels=branches, autopct="%1.1f%%", startangle=140)

plt.title("Revenue Contribution by Branch")

plt.savefig("pie\_chart\_revenue\_contribution.png")

plt.show()

# 4. Employee Satisfaction vs. Branch Profit

branch\_profits = annual\_revenue\_per\_branch - monthly\_expenses.sum(axis=0)

plt.figure(figsize=(10, 6))

plt.scatter(employee\_satisfaction, branch\_profits, c='orange', edgecolors='k', s=100)

plt.title("Employee Satisfaction vs. Branch Profit")

plt.xlabel("Employee Satisfaction")

plt.ylabel("Profit ($)")

plt.savefig("scatter\_plot\_satisfaction\_vs\_profit.png")

plt.show()

# 5. Dashboard with Multiple Visualizations

fig, axes = plt.subplots(2, 2, figsize=(15, 10))

axes[0, 0].plot(monthly\_revenue[:, np.argmax(annual\_revenue\_per\_branch)], marker='o')

axes[0, 0].set\_title("Top Branch Revenue Trend")

axes[0, 1].bar(branches, annual\_revenue\_per\_branch, color='skyblue')

axes[0, 1].set\_title("Annual Revenue by Branch")

axes[1, 0].pie(annual\_revenue\_per\_branch, labels=branches, autopct="%1.1f%%")

axes[1, 0].set\_title("Revenue Contribution")

axes[1, 1].scatter(employee\_satisfaction, branch\_profits, c='orange', edgecolors='k')

axes[1, 1].set\_title("Satisfaction vs Profit")

plt.tight\_layout()

plt.savefig("dashboard.png")

plt.show()

# DataFrame Operations

data = pd.DataFrame({

"Branch": branches,

"Annual Revenue": annual\_revenue\_per\_branch,

"Total Expenses": monthly\_expenses.sum(axis=0),

"Employee Satisfaction": employee\_satisfaction

})

data["Net Income"] = data["Annual Revenue"] - data["Total Expenses"]

data["Profit Margin"] = (data["Net Income"] / data["Annual Revenue"]) \* 100

filtered\_data = data[data["Annual Revenue"] > 750\_000]

sorted\_data = data.sort\_values(by="Employee Satisfaction", ascending=False)

data.to\_csv("datamart\_analysis.csv", index=False)

**Conclusion**

The **DataMart Annual Operations Analysis** successfully demonstrates how data-driven insights can enhance decision-making processes within a retail organization. By analyzing financial data, employee satisfaction, and branch performance, we identified key trends and actionable opportunities for improvement. The categorization of branches based on revenue and satisfaction metrics highlighted areas requiring attention, while the visualizations provided a clear and intuitive representation of the company's operational health. The calculated financial metrics, such as profit margins and net income, offered a detailed understanding of branch profitability.

This project underscores the importance of leveraging Python's powerful libraries, such as **Pandas**, **NumPy**, **Matplotlib**, and **Seaborn**, to process and visualize large datasets effectively. The resulting insights can guide strategic decisions aimed at optimizing revenue generation, improving employee satisfaction, and maintaining financial stability. With the provided scripts and visualizations, **DataMart** can continually monitor performance and adapt to evolving business needs, ensuring sustainable growth and competitive advantage in the retail industry.

**References**

1. NumPy Documentation  
   <https://numpy.org/doc/>
2. Pandas Documentation  
   <https://pandas.pydata.org/docs/>
3. Matplotlib Documentation  
   <https://matplotlib.org/stable/contents.html>
4. Seaborn Documentation  
   <https://seaborn.pydata.org/>
5. Python Official Documentation  
   <https://docs.python.org/3/>
6. NumPy Random Module  
   <https://numpy.org/doc/stable/reference/random/>