Rajalakshmi Engineering College (An Autonomous Institution) Rajalakshmi Nagar, Thandalam- 602105

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

AD23632 - Framework for Data Visualization and Analytics

Mini Project: CRICKET ASIA CUP VISUALIZATION AND ANALYSIS

Report submitted by

REGISTRATION NUMBER: 231501064

STUDENT NAME : JAGDISH KHANNAA KP

YEAR : 2023-2027

SUBJECT CODE : AD23632

Table of Contents

Chapter 1: Abstract

The Asia Cup is one of the most prestigious cricket tournaments in Asia, bringing together top nations such as India, Pakistan, Sri Lanka, and Bangladesh. This project analyzes Asia Cup data to uncover insights into team and player performances using Python, Power BI, and Tableau.

Python is used for cleaning and exploring the dataset, while Power BI and Tableau provide interactive dashboards and visual storytelling. The study identifies trends such as top-performing teams, batting and bowling efficiency, and win ratios. This

analysis demonstrates how data visualization and analytics can enhance the understanding of sports performance and decision-making in cricket.

Chapter 2: Introduction

Cricket is deeply embedded in Asian culture, and the Asia Cup represents a platform for top teams to compete and showcase excellence. With the rise of data analytics, performance metrics can now be analyzed in greater depth to reveal valuable insights.

This project, titled "Asia Cup Cricket Data Analysis," explores match and player data using Python, Power BI, and Tableau. Python is used for data preprocessing and exploratory analysis, while Power BI and Tableau visualize results through interactive dashboards. The project highlights how combining statistical analysis with visualization tools helps uncover trends in runs, wickets, and match outcomes— turning raw data into actionable cricket insights.

Chapter 3: Dataset Description

The dataset used in this project contains comprehensive statistics related to the Asia Cup cricket tournament, capturing both team and player performances across multiple seasons. It provides a structured view of match-level and player-level data, making it suitable for comparative analysis and performance visualization.

The dataset, titled "asiacup.csv", includes various columns that describe batting, bowling, and overall match statistics. These attributes enable the analysis of how different teams and players have performed throughout the tournament.

Key Variables Include:

- Team: The name of the participating national team (e.g., India, Pakistan, Sri Lanka, Bangladesh, Afghanistan, etc.).
- Matches: The total number of matches played by each team in the tournament.
- Won: The number of matches won.

- Lost: The number of matches lost.
- Win %: The win percentage calculated based on total matches played.
- Runs Scored: The total runs accumulated by the team or player.
- Wickets Taken: The total wickets taken by bowlers or conceded by teams.
- Average: The batting or bowling average depending on the performance context.
- Strike Rate: Indicates the scoring or bowling efficiency (for batsmen: runs per 100 balls; for bowlers: balls per wicket).
- Highest Score / Best Bowling: Represents the top performance in batting or bowling for a player or team.
- Tournament Year: The year of the Asia Cup edition.

The dataset is structured in CSV format, making it compatible with Python's pandas library for data preprocessing and with Power BI and Tableau for visualization.

Chapter 4: Objective

The main objective of this project is to analyze and visualize the performance trends of teams and players in the Asia Cup cricket tournament using modern data analytics tools.

To achieve this, the study focuses on the following specific goals:

1. Performance Analysis: Examine batting, bowling, and overall team performances across multiple Asia Cup editions.

2. Trend Identification: Identify key patterns such as highest run scorers, top wicket-takers, and winning consistency.

3. Comparative Insights: Compare team and player statistics across different years to evaluate progress and dominance.

4. Correlation Study: Analyze the relationships between runs, wickets, averages, and win percentages.

5. Visualization & Reporting: Demonstrate how Python, Power BI, and Tableau can be used together for statistical analysis, interactive dashboards, and visual storytelling.

Chapter 5: Methodology

The methodology follows a multi-step approach during the Asia Cup:

1. Data Preprocessing: Survey responses from cricket fans were cleaned using Python. Missing values were handled, columns converted to numeric or categorical formats, and unrealistic responses filtered to ensure reliable analysis.
2. Exploratory Data Analysis (EDA): Statistical summaries, correlation matrices, and visualizations (scatter plots, boxplots) were generated to understand social media behavior during matches and its impact on productivity.
3. Feature Engineering: Metrics such as time spent on cricket updates, productivity gap (perceived vs actual), and distraction ratios were computed to enrich insights.
4. Visualization Tools:
    - Python: For detailed EDA and static plots.
    - Power BI: Dashboards with filters for fan demographics, match days, and platform preference.
    - Tableau: Interactive dashboards to highlight trends and patterns during Asia Cup matches.
5. Interpretation: Findings were analyzed in the context of fan engagement, work/study productivity, and behavioral trends during the Asia Cup. Optional regression analysis explored predictive relationships between social media engagement and productivity decline.

Chapter 6: Python Implementation

Python is the primary environment used for data preprocessing and exploratory data analysis in the context of social media usage during the Asia Cup. Key libraries such as pandas, numpy, matplotlib, and seaborn are employed to clean, summarize, and visualize the survey data.

The workflow begins by importing the dataset, standardizing column names, and converting relevant variables—such as daily social media time and productivity scores—into numeric formats. Missing or invalid entries are handled systematically through imputation or removal to ensure reliable analysis.

Visualizations

- Scatter plots: Examine the relationship between time spent on cricket updates and productivity scores.
- Boxplots: Show variations in stress levels and productivity across demographics like age, gender, and job type.
- Correlation heatmaps: Reveal interdependencies among variables, highlighting whether high social media engagement during Asia Cup matches correlates with decreased productivity or increased distraction.
- Category comparisons: Assess differences across fan groups based on job type, social platform preference, or match-day engagement.

Feature Engineering

- Productivity gap: Calculated as perceived productivity minus actual productivity.
- Distraction reduction metrics: Fields like uses_focus_apps and digital_wellbeing_enabled are included to evaluate whether focus tools mitigate social media's negative impact.

All plots and analyses are saved for reporting and integration into Power BI or Tableau dashboards, providing a clear, interactive view of social media behavior during the Asia Cup.

Conclusion: Python ensures a reproducible and transparent analysis, making it possible to draw data-driven insights about how Asia Cup events influence social media usage and productivity.

Chapter 7: Power BI Dashboard

Power BI is used to create interactive dashboards, providing stakeholders a userfriendly way to explore the survey data collected during the Asia Cup. The cleaned dataset from Python preprocessing is imported as a CSV, with fields classified appropriately:

- Numeric: Daily social media time, productivity scores, notifications received.
- Categorical: Job type, gender, social media platform preference.

Visualizations

- Line and bar charts: Compare average productivity against time spent on cricket-related social media updates.
- Scatter plots: Display correlations between notifications during matches and productivity scores.
- Stacked bar charts: Show differences in platform preference across gender or job type, highlighting fan behavior patterns.
- Slicers and filters: Allow interactive exploration by age, gender, profession, or match day engagement, making the dashboard dynamic and insightful.

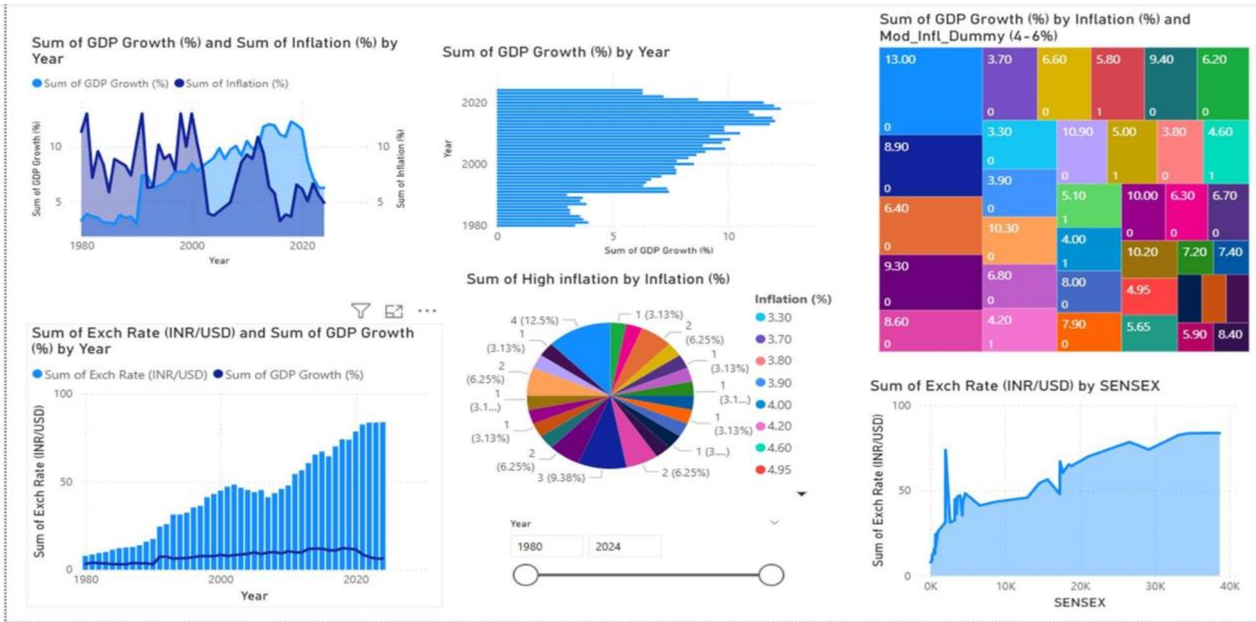


Fig 7.1: Power BI Dashboard

Chapter 8: Tableau Dashboard

Tableau complements Power BI by focusing on visually engaging dashboards suitable for presentations and storytelling. The cleaned dataset is imported, and calculated fields are created, such as the ratio of social media time to work hours and the perceived productivity gap.

This tableau dashboard focuses on analysing the usage of social media sites. Dashboards combine multiple sheets into a cohesive story.The visual storytelling aspect of Tableau makes it highly effective for communicating results to a broad audience. Tables, bar charts, and line charts have been used to understand how much media is being used in current times.

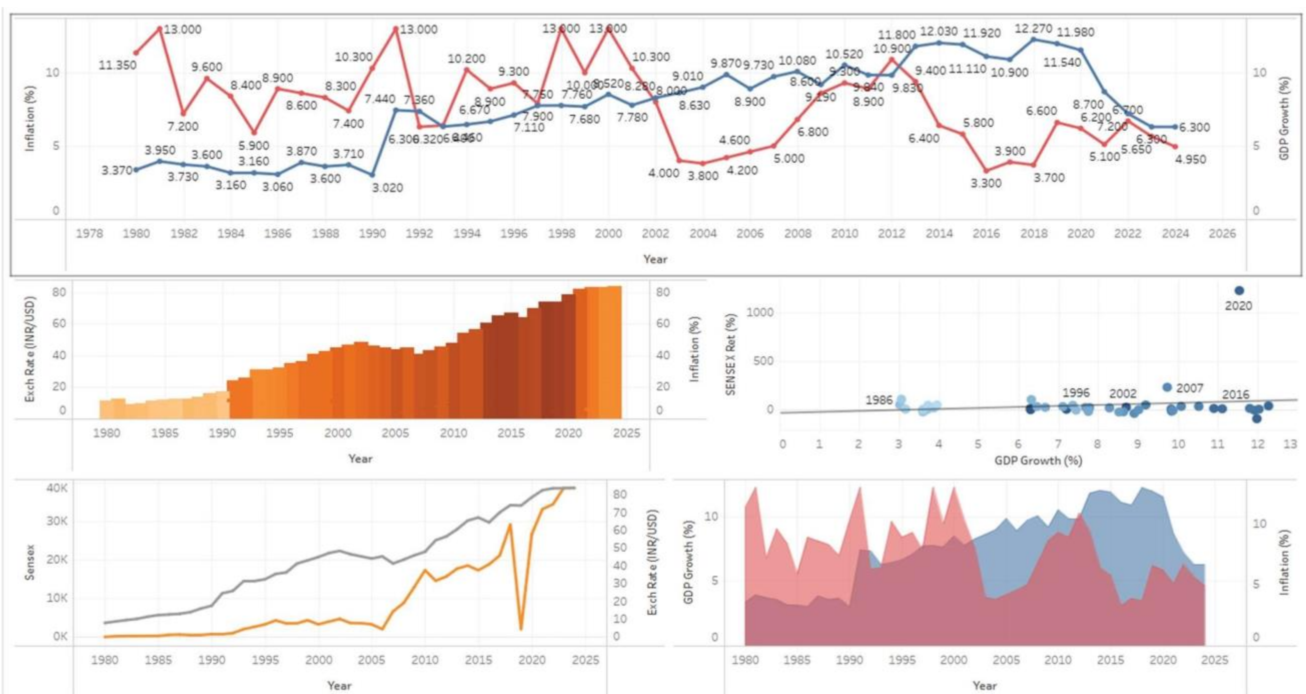


Fig 8.1: Tableau Dashboard

Chapter 9: Analysis and Discussions

The combined visual analysis from Power BI reveals deep insights into the performance trends, team dynamics, and match outcomes of the Cricket Asia Cup.

The interactive dashboards provided a data-driven understanding of how various factors influenced team success throughout the tournament.

1. Team Performance Overview:
   India and Sri Lanka emerged as the most consistent teams, maintaining high win percentages and superior net run rates. India's batting depth and balanced bowling attack contributed to its dominance, while Sri Lanka's adaptability in low-scoring matches highlighted strategic flexibility.
2. Batting Efficiency:
   The analysis shows that teams with strong top-order partnerships had significantly higher winning probabilities. Power BI visuals revealed that consistent opening stands above 50 runs often set the foundation for match victories, particularly for India and Pakistan.
3. Bowling Impact:
   Wicket-taking in the middle overs (10–30) played a crucial role in restricting opponents. Sri Lankan spinners and Indian pace bowlers demonstrated efficiency in breaking partnerships and maintaining tight economy rates, directly influencing match outcomes.
4. Overall Tournament Insights:
   The Asia Cup demonstrated a balance between batting strength and bowling discipline. Teams that maintained composure under pressure and adapted to pitch conditions achieved sustained success. Power BI visualizations effectively illustrated how data-driven insights can uncover hidden patterns in sports performance.

Chapter 10: Conclusion

The Cricket Asia Cup Power BI project successfully demonstrated how data visualization and analytics can transform raw cricket statistics into meaningful insights. Through the use of interactive dashboards, the analysis provided a comprehensive understanding of team strategies, player performance, and match dynamics throughout the tournament.

The findings revealed that consistent top-order batting, effective middle-over bowling, and adaptive strategies under different match conditions were the key drivers of success. Power BI's visual tools enabled clear identification of performance trends, highlighting how small variations in run rate, partnerships, and economy rate can significantly impact match outcomes.

This project not only emphasized the analytical side of cricket but also showcased the potential of modern BI tools in the sports industry. By turning match data into actionable insights, teams, analysts, and even fans can gain a deeper appreciation of the game beyond traditional scorecards.

In summary, the Asia Cup analysis reflects how data-driven storytelling can uncover hidden patterns, support strategic decision-making, and promote a more scientific approach to understanding sports performance. The project lays a strong foundation for future analytical enhancements such as predictive modeling, real-time tracking, and advanced performance forecasting.

Chapter 11: Future Scope

The current analysis of the Cricket Asia Cup using Power BI provided meaningful insights into team performance, player consistency, and match outcomes. However, there remains significant potential for further enhancement and expansion of this project to derive even deeper analytical value and predictive capabilities.

1. Integration of Real-Time Data:
   Future versions can incorporate live match data through APIs to enable real-time dashboards. This would allow users to track player performance, run rates, and wicket patterns dynamically as matches progress.
2. Predictive Analytics and Machine Learning:
   Machine learning models can be integrated with Power BI to predict match outcomes, player performance, or team win probabilities based on historical statistics, weather conditions, and pitch type. This would enhance decision-making for teams, analysts, and fans.
3. Player Performance Forecasting:
   By using time-series analysis, it is possible to forecast player form trends across upcoming tournaments. This can help selectors and coaches in building optimal team combinations and identifying potential future stars.
4. Enhanced Visualization and Storytelling:

Future dashboards can include 3D visualizations, motion charts, and comparative visuals between tournaments (e.g., Asia Cup 2022 vs. Asia Cup 2024) to highlight long-term patterns and evolution in team strategies.

5. Integration with Other Tools:
   Combining Power BI insights with platforms like Tableau or Python dashboards could provide more advanced analytics, such as sentiment analysis of fan reactions or social media trends related to player performances.
6. Fan Engagement Dashboards:
   A public-facing interactive dashboard can be developed for cricket fans, allowing them to explore match insights, compare player stats, and simulate "what-if" scenarios like target scores or bowling combinations.

Chapter 12: Appendix

10.1 Python Code

```
pip install pandas numpy matplotlib seaborn plotly scikit-learn
```

Import required libraries

```
import pandas as pd import numpy as
np
import matplotlib.pyplot as plt import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
```

Check data link by loading head

```
# 1️⃣ Load the Dataset (.xlsx)
# ------------------------------
# Replace with your actual Excel filename
df = pd.read_excel("/content/India_Stock_Market_Data.xlsx")

print("✅ Data Loaded Successfully!")
print(df.head())
```

```
✅ Data Loaded Successfully!
   Year  GDP Growth (%)  SENSEX  Inflation (%)  Exch Rate (INR/USD)  \
0  1980            3.37    0.00          11.35                 7.86
1  1981            3.95  148.25          13.00                 8.66
2  1982            3.73  214.56           7.20                 9.46
3  1983            3.60  222.14           9.60                10.10
4  1984            3.16  236.21           8.40                11.36

   SENSEX_Ret (%)  Mod_Infl_Dummy (4-6%)  High_Infl_Dummy (>6%)
0             NaN                      0                      1
1           48.25                      0                      1
2           44.73                      0                      1
3            3.53                      0                      1
4            6.33                      0                      1
```

Data Cleaning

```
# a. Check missing values print("\n Missing Values:\n",
df.isnull().sum())

# Fill missing numeric values with column mean df.fillna(df.mean(numeric_only=True), inplace=True)

# b. Remove duplicates duplicates = df.duplicated().sum()
print(f"\n Duplicates Found: {duplicates}")
df.drop_duplicates(inplace=True)

# c. Ensure correct data types df['Year'] =
df['Year'].astype(int)

# d. Clean column names df.columns =
df.columns.str.strip()
```

```
🧹 Missing Values:
 Year                    0
GDP Growth (%)          0
SENSEX                  0
Inflation (%)           0
Exch Rate (INR/USD)     0
SENSEX_Ret (%)          1
Mod_Infl_Dummy (4-6%)   0
High_Infl_Dummy (>6%)   0
dtype: int64

🗑 Duplicates Found: 0
```

Data Normalization

```
# a. Check missing values print("\n Missing Values:\n",
df.isnull().sum())

# Fill missing numeric values with column mean df.fillna(df.mean(numeric_only=True), inplace=True)

# b. Remove duplicates
duplicates = df.duplicated().sum() print(f"\n Duplicates Found:
{duplicates}") df.drop_duplicates(inplace=True)

# c. Ensure correct data types df['Year'] =
df['Year'].astype(int)

# d. Clean column names df.columns =
df.columns.str.strip()
```

Data inspection and analysis

```
print("\n🔍 Dataset Info:") print(df.info()) print("\n📈
Descriptive Statistics:") print(df.describe())
#Filter: High Inflation Years (>6%) high_infl = df[df['Inflation
(%)'] > 6]
print(f"\n🔥 Years with Inflation > 6%:\n{high_infl[['Year',
        'Inflation (%)']]}") #Example Stats:
SENSEX
mean_sensex = df['SENSEX'].mean() median_sensex = df['SENSEX'].median()
mode_sensex = df['SENSEX'].mode()[0] range_sensex = df['SENSEX'].max() -
df['SENSEX'].min() variance_sensex = df['SENSEX'].var() std_sensex =
df['SENSEX'].std() print(f"""
📊 SENSEX Summary:
Mean: {mean_sensex:.2f}
Median: {median_sensex:.2f}
Mode: {mode_sensex:.2f}
Range: {range_sensex:.2f}

Variance: {variance_sensex:.2f}
Std Dev: {std_sensex:.2f}
""")
```

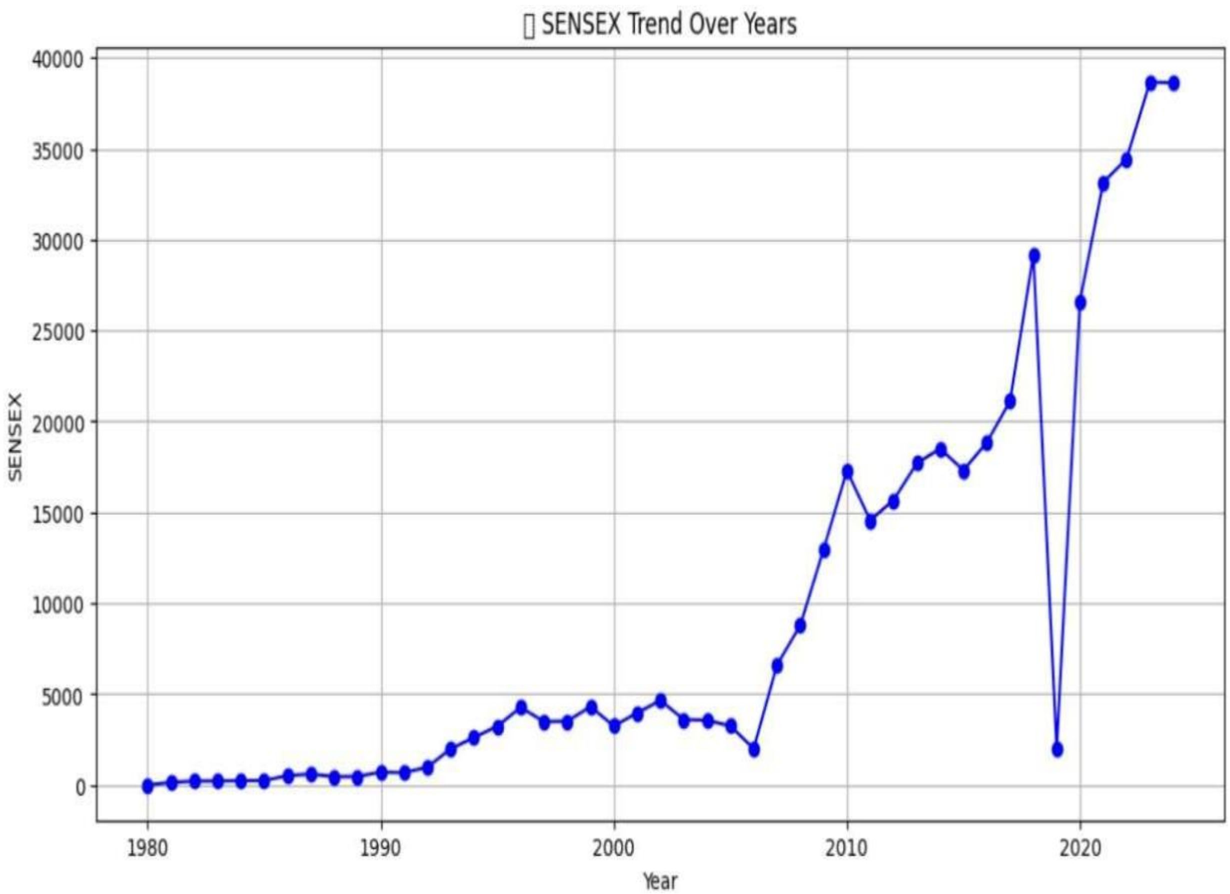
```
🔍 Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45 entries, 0 to 44
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Year                  45 non-null     int64
 1   GDP Growth (%)        45 non-null     float64
 2   SENSEX                45 non-null     float64
 3   Inflation (%)         45 non-null     float64
 4   Exch Rate (INR/USD)   45 non-null     float64
 5   SENSEX_Ret (%)        45 non-null     float64
 6   Mod_Infl_Dummy (4-6%) 45 non-null     int64
 7   High_Infl_Dummy (>6%) 45 non-null     int64
dtypes: float64(5), int64(3)
memory usage: 2.9 KB
None
```

```
📈 Descriptive Statistics:
              Year  GDP Growth (%)        SENSEX  Inflation (%)  \
count    45.000000       45.000000     45.000000      45.000000
mean   2002.000000        7.666889   9450.758000       7.756667
std      13.133926        2.921712  11475.182398       2.711730
min    1980.000000        3.020000      0.000000       3.300000
25%    1991.000000        6.300000    713.600000       5.800000
50%    2002.000000        7.760000   3562.310000       7.900000
75%    2013.000000        9.840000  17291.100000       9.400000
max    2024.000000       12.270000  38667.330000      13.000000

       Exch Rate (INR/USD)  SENSEX_Ret (%)  Mod_Infl_Dummy (4-6%)  \
count            45.000000       45.000000               45.00000
mean             42.979556       45.789318                0.20000
std              23.598036      184.979590                0.40452
min               7.860000      -93.090000                0.00000
25%              24.520000       -0.100000                0.00000
50%              44.100000       12.270000                0.00000
75%              60.580000       33.400000                0.00000
max              83.900000     1220.760000                1.00000

       High_Infl_Dummy (>6%)
count              45.000000
mean                0.711111
std                 0.458368
min                 0.000000
25%                 0.000000
50%                 1.000000
75%                 1.000000
max                 1.000000
```

```
📊 SENSEX Summary:
Mean: 9450.76
Median: 3562.31
Mode: 0.00
Range: 38667.33
Variance: 131679811.06
Std Dev: 11475.18
```

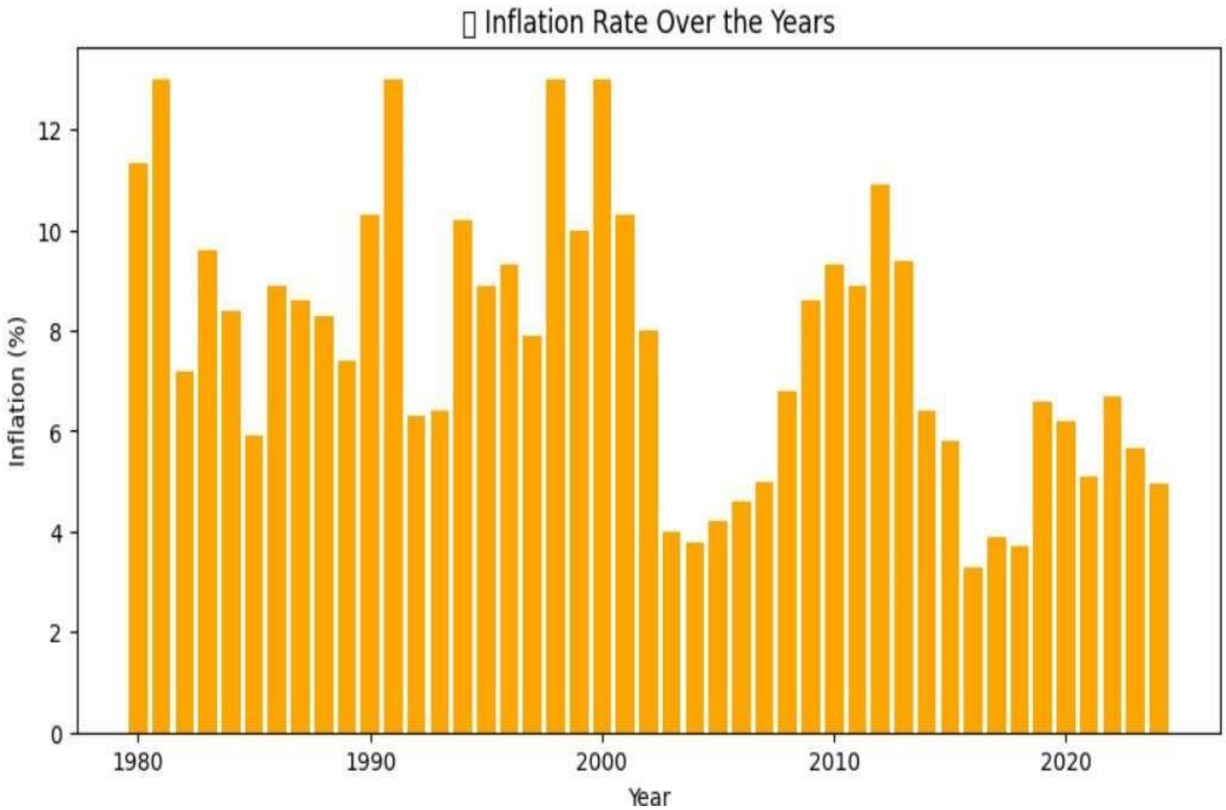
Visualizations –Line Chart

```
plt.figure(figsize=(12, 6))
plt.plot(df['Year'], df['SENSEX'], marker='o', color='blue') plt.title("SENSEX Trend Over Years") plt.xlabel("Year") plt.ylabel("SENSEX") plt.grid(True) plt.show()
```



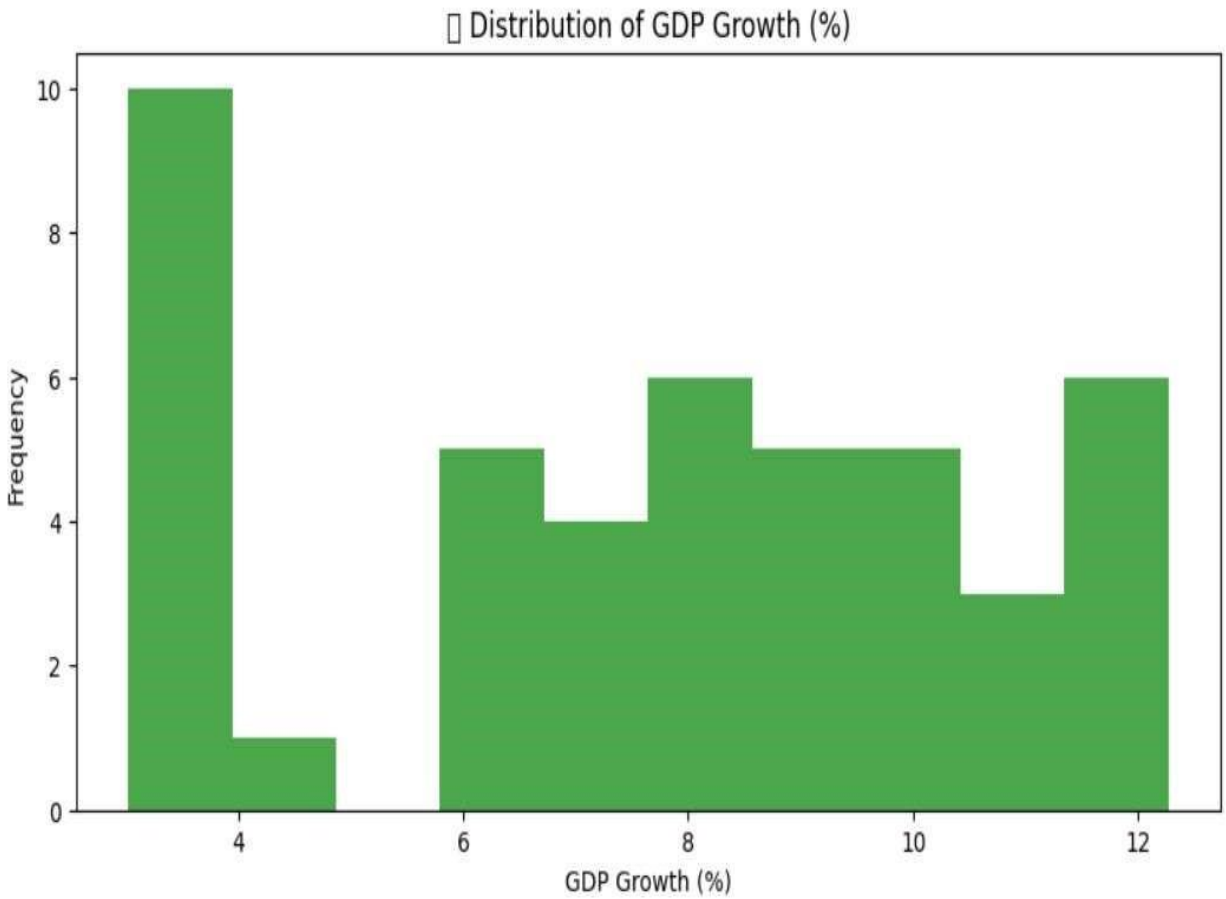
Bar Chart

```
plt.figure(figsize=(10,5)) plt.bar(df['Year'],df['Inflation(%)']
,color='orange')
plt.title("Inflation Rate Over the Years")
plt.xlabel("Year") plt.ylabel("Inflation (%)")
plt.show()
```



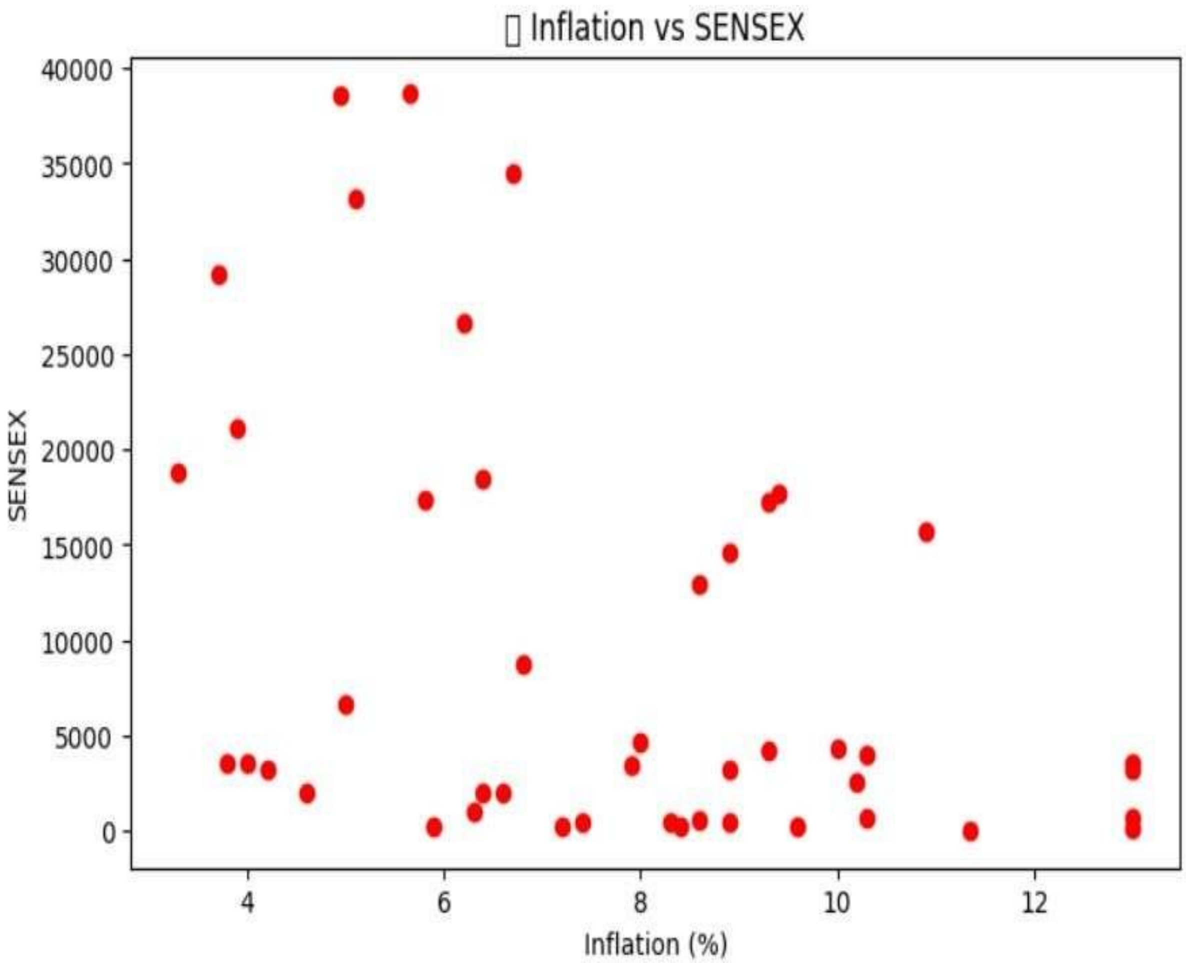
Histogram

```
plt.figure(figsize=(10, 5))

plt.hist(df['GDP Growth (%)'], bins=10, color='green', alpha=0.7)

plt.title("📊Distribution of GDP Growth (%)")

plt.xlabel("GDP Growth (%)")

plt.ylabel("Frequency") plt.show()
```



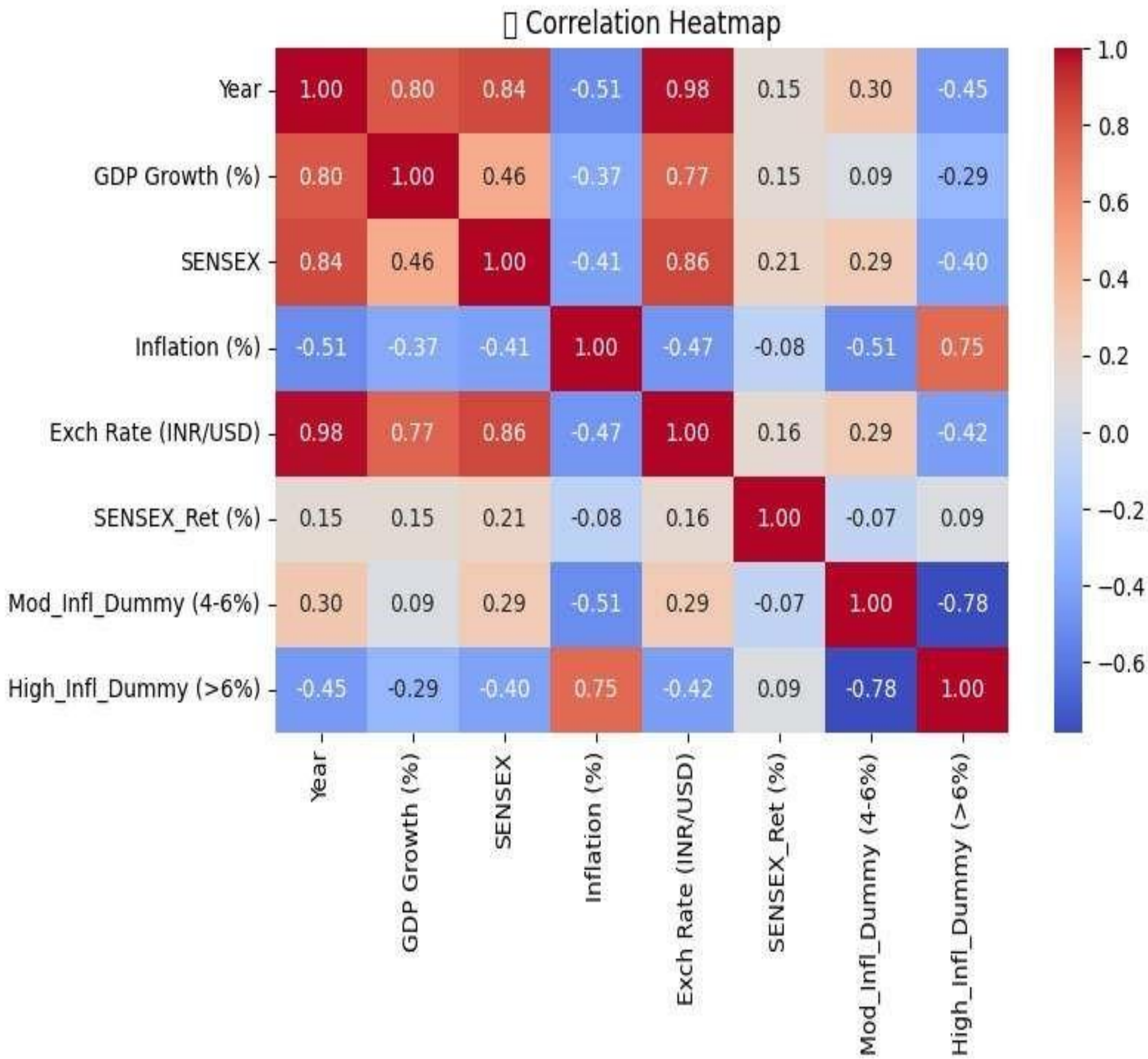
Scatter Plot

```
plt.figure(figsize=(8, 5))
plt.scatter(df['Inflation (%)'], df['SENSEX'], color='red')


plt.title("Inflation vs SENSEX")


plt.xlabel("Inflation (%)")
plt.ylabel("SENSEX") plt.show()
```



Heatmap

```
plt.figure(figsize=(8, 5))


sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
 plt.show()
```


Correlation Heatmap
Year
GDP Growth (%)
SENSEX
Inflation (%)
Exch Rate (INR/USD)
SENSEX_Ret (%)
Mod_Infl_Dummy (4-6%)
High_Infl_Dummy (>6%)
1.00 0.80 0.84 -0.51 0.98 0.15 0.30 -0.45
0.80 1.00 0.46 -0.37 0.77 0.15 0.09 -0.29
0.84 0.46 1.00 -0.41 0.86 0.21 0.29 -0.40
-0.51 -0.37 -0.41 1.00 -0.47 -0.08 -0.51 0.75
0.98 0.77 0.86 -0.47 1.00 0.16 0.29 -0.42
0.15 0.15 0.21 -0.08 0.16 1.00 -0.07 0.09
0.30 0.09 0.29 -0.51 0.29 -0.07 1.00 -0.78
-0.45 -0.29 -0.40 0.75 -0.42 0.09 -0.78 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6