AI MIDSEM – 1 EXAMINATION

NAME – PRATEEK BALIYAN

BRANCH – CSEAI

SECTION – C

ROLL NUMBER (UNI) – 202401100300177

ROLL NUMBER (CLASS) – 30

TOPIC NAME – TRAFFIC LIGHT CONTROL SYSTEM

INTRODUCTION

Traffic data analysis is crucial for understanding web user behaviour, optimizing website performance, and improving user engagement. This report examines traffic data over a 20-day period, focusing on key metrics such as Page Views, Unique Visitors, and Bounce Rate.

The goal of this study is to:

Identify traffic trends over time.

Analyse the relationship between Page Views and Bounce Rate.

Provide insights for optimizing web performance.

Using Python, pandas, matplotlib, and seaborn, the dataset was processed to extract meaningful patterns and visualizations. The findings help in understanding visitor engagement and website efficiency.

METHODOLOGY

Methodology for the Given Code

1. Data Preprocessing (AI Foundation)

Handling Missing Values:

Checking for missing values using df.isnull().sum() is a crucial preprocessing step in AI to ensure data quality.

Feature Engineering:

Extracting the hour from the timestamp (df['timestamp'].dt.hour) is a common AI technique to create meaningful features for time-series models.

2. Exploratory Data Analysis (EDA)

Statistical Analysis:

Using df.describe() helps understand the data distribution, an essential step before applying machine learning models.

Time-Series Analysis:

Grouping data by hour (df.groupby('hour').size()) is a fundamental step in predictive modeling and anomaly detection in AI.

3. Visualization (AI-Powered Insights)

Seaborn & Matplotlib Plots:

These visualizations help identify trends, outliers, and patterns, which are crucial for training AI models like time-series forecasting (e.g., ARIMA, LSTMs) or anomaly detection models (e.g., Isolation Forests, Autoencoders).

CODE

import pandas as pd

import matplotlib.pyplot as plt

# Load the CSV file

file\_path = "/content/traffic\_data.csv"  # Update with the correct path if needed

df = pd.read\_csv(file\_path)

print("Dataset Information:\n")

print(df.info())

# Show first few rows

print("\nFirst 5 Rows:\n", df.head())

# Check for missing values

print("\nMissing Values:\n", df.isnull().sum())

# Basic statistics

print("\nStatistical Summary:\n", df.describe())

# Time-based traffic analysis (Assuming there's a 'timestamp' column)

if 'timestamp' in df.columns:

    df['timestamp'] = pd.to\_datetime(df['timestamp'])

    df['hour'] = df['timestamp'].dt.hour

    traffic\_by\_hour = df.groupby('hour').size()

    plt.figure(figsize=(10, 5))

    sns.lineplot(x=traffic\_by\_hour.index, y=traffic\_by\_hour.values, marker='o')

    plt.title('Traffic Volume by Hour')

    plt.xlabel('Hour of Day')

    plt.ylabel('Traffic Count')

    plt.grid()

    plt.show()

# Convert 'Date' column to datetime format

df['Date'] = pd.to\_datetime(df['Date'])

# Create the plot

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

plt.plot(df['Date'], df['PageViews'], label='Page Views', marker='o', linestyle='-')

plt.plot(df['Date'], df['UniqueVisitors'], label='Unique Visitors', marker='s', linestyle='--')

# Formatting

plt.xlabel("Date")

plt.ylabel("Count")

plt.title("Website Traffic Over Time")

plt.xticks(rotation=45)

plt.legend()

plt.grid(True)

# Show the plot

plt.show()

OUTPUT

