Analyzing Website Traffic Data

Name : Ayush Prasad

Roll no: 2020401100300086

Branch : CSE-AI

Sec: B

Introduction:

This report provides a comprehensive analysis of website traffic data using Python. The goal of this analysis is to understand website visitor behavior, identify peak traffic times, and derive insights that can help optimize website performance. This is achieved through data preprocessing, exploratory data analysis (EDA), and visualization techniques

---

Data Collection

The dataset used in this analysis was obtained and uploaded using Google Colab's file upload feature. The file was loaded into a Pandas DataFrame for further processing and analysis. The dataset includes key metrics such as page views, unique visitors, and bounce rates, which help measure website engagement and performance.

Data Preprocessing

To ensure the accuracy and usability of the dataset, several preprocessing steps were taken:

- **Loading the Data:** The dataset was read into a Pandas DataFrame.
- **Data Structure Examination:** The dataset's structure, including column types and missing values, was analyzed using `df.info()`.

- **Initial Data Review:** The first few rows of the dataset were displayed to understand its contents.
- **Datetime Conversion:** If the dataset contained a `timestamp` column, it was converted to a proper datetime format using Pandas.
- **Feature Engineering:** Additional features, such as `hour` and `day`, were extracted from the timestamp column to facilitate deeper analysis

---

Exploratory Data Analysis (EDA)

EDA was conducted to uncover patterns and trends in the website traffic data:

- **Dataset Information:** The data types, missing values, and structure of the dataset were examined.
- **First Few Rows:** A sample of the dataset was reviewed to understand its composition.
- **Summary Statistics:** Descriptive statistics, such as mean, standard deviation, minimum, and maximum values, were calculated for numerical columns.
- **Data Distribution:** Histograms and other plots were used to visualize the spread of key metrics like page views and bounce rates.

Traffic Trend Analysis

Understanding traffic trends over time is crucial for identifying patterns in user activity:

- A **daily traffic trend plot** was generated using Matplotlib to visualize fluctuations in the number of visits.
- The data was **resampled on a daily basis**, allowing for a clearer view of overall trends.
- Spikes and dips in traffic were analyzed to determine possible causes, such as marketing campaigns, holidays, or technical issues.

Peak Hours Analysis

Peak hours indicate the times when website traffic is at its highest:

- A count plot was generated using Seaborn to analyze the distribution of website traffic by hour.
- The busiest hours were identified, providing valuable insights into user activity patterns.
- These insights can be used to optimize server performance, schedule content updates, and improve user engagement strategies.

Visualization & Insights

- **Daily Traffic Trend:** A time-series plot was created to illustrate how website traffic fluctuates over time. This helps in understanding seasonal patterns, sudden spikes, and general trends.
- **Peak Hour Distribution:** A bar chart was generated to visualize the intensity of user visits at different times of the

day. This insight is useful for optimizing content delivery and server resource allocation.

Conclusion

The analysis of website traffic data provided valuable insights into user behavior and site engagement. Key takeaways include:

- Identification of peak traffic hours, helping in optimizing website operations.
- Understanding fluctuations in daily visits, allowing businesses to plan their marketing strategies accordingly.
- Detection of anomalies in traffic patterns that could indicate technical issues or special events.

Recommendations

Based on the insights derived from this analysis, the following recommendations are suggested:

- **Optimize Server Performance:** Allocate more resources during peak hours to maintain a seamless user experience.
- **Strategic Content Scheduling:** Publish important content and updates when traffic is highest to maximize visibility and engagement.
- **Further Data Segmentation:** Analyze traffic by source, user demographics, and device type for a more detailed understanding of visitor behavior.

- **Improve Bounce Rate:** Investigate pages with high bounce rates and optimize their content or design to improve user retention.

References & Credits

The following resources were used for this analysis:

- Dataset: Uploaded via Google Colab from local storage.
- Libraries Used: Pandas, Matplotlib, Seaborn, Google Colab's file upload module.
- Colab Notebook: Python script executed in Google Colab for data analysis.
- Visualizations: Created using Matplotlib and Seaborn.

Google Colab Link

To access the Google Colab notebook containing the analysis and code execution, click the link below: Google Colab Notebook *https://colab.research.google.com/drive/1yCHV7vdG5D_lG_Qv101Yu-PxAMoqicZI?usp=sharing*

This report serves as a foundational analysis of website traffic data and can be expanded with more advanced techniques such as predictive modeling and user segmentation to gain deeper insights.

CODE:

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from google.colab import files

import io


# Upload file using Colab's file upload

uploaded = files.upload()

file_name = list(uploaded.keys())[0]  # Get the uploaded file name

print("Using file:", file_name)


# Load the data

df = pd.read_csv(io.StringIO(uploaded[file_name].decode('utf-8')))


# Display basic information

print("Dataset Info:")

print(df.info())
```

```
print("\nFirst few rows:")
print(df.head())

# Convert date/time columns if applicable
if 'timestamp' in df.columns:
    df['timestamp'] = pd.to_datetime(df['timestamp'])
    df['hour'] = df['timestamp'].dt.hour
    df['day'] = df['timestamp'].dt.date

# Traffic trend over time
if 'timestamp' in df.columns:
    df.set_index('timestamp', inplace=True)
        df.resample('D').size().plot(title='Daily Traffic Trend', figsize=(10, 5))
    plt.xlabel('Date')
    plt.ylabel('Number of Visits')
    plt.show()

# Peak hours analysis
if 'hour' in df.columns:
```

```
    plt.figure(figsize=(10, 5))
    sns.countplot(x='hour', data=df, palette='viridis')
    plt.title('Traffic Distribution by Hour')
    plt.xlabel('Hour of the Day')
    plt.ylabel('Number of Visits')
    plt.show()

# Display summary statistics
print("\nSummary Statistics:")
print(df.describe())
```

Output:

```
Choose files traffic_data.csv
• traffic_data.csv(text/csv) - 820 bytes, last modified: 11/03/2025 - 100% done
Saving traffic_data.csv to traffic_data (3).csv
Using file: traffic_data (3).csv
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            20 non-null     object
 1   PageViews       20 non-null     int64
 2   UniqueVisitors  20 non-null     int64
 3   BounceRate      20 non-null     float64
dtypes: float64(1), int64(2), object(1)
memory usage: 772.0+ bytes
None

First few rows:
         Date  PageViews  UniqueVisitors  BounceRate
0  2024-01-01        828            1261   54.420009
1  2024-01-02       7065            4225   31.583887
2  2024-01-03       5861            3286   68.284703
3  2024-01-04       7163             651   60.203175
4  2024-01-05       9432             548   37.963247

Summary Statistics:
        PageViews  UniqueVisitors  BounceRate
count    20.00000        20.00000   20.000000
mean   5533.20000      2435.05000   49.150658
std    2595.96585      1383.40109   15.286241
min     828.00000       518.00000   28.581849
25%    3218.50000      1115.25000   37.609458
50%    6405.00000      2466.50000   49.061288
75%    7288.75000      3696.25000   60.163514
max    9432.00000      4459.00000   79.981676
```