**Data Analytics with Cognos**

**Project Tittle: Website Traffic Analysis**

**Phase 5: Project Documentation & Submission**

****

**Website Traffic Analysis**

**Introduction:**

* Website traffic analysis is the process of monitoring, assessing, and interpreting the data related to the visitors who land on a website. It involves collecting a wealth of information about user behavior, such as which pages they visit, how long they stay, and how they arrived at the site. This data provides invaluable insights into the effectiveness of a website's content, design, and marketing efforts.
* Why is website traffic analysis important? Simply put, it empowers website owners, marketers, and webmasters to make data-driven decisions that can have a profound impact on their online presence.
* Whether you're a business owner looking to increase your online sales, a blogger seeking to grow your readership or a non-profit organization aiming to reach a wider audience, website traffic analysis is the key to optimizing your web presence and achieving your goals.
* In this guide, we'll delve into the world of website traffic analysis, exploring the tools, techniques, and strategies that can help you make the most of your online presence.

**Project’s objective:**

1. Analysis Objectives: Define the key insights you want to extract from the website traffic data, such as identifying popular pages, traffic trends, and user engagement metrics.
2. Data Collection: Determine the data sources and methods for collecting website traffic data, including page views, unique visitors, referral sources, and more.
3. Visualization: Plan how to visualize the insights using IBM Cognos to create meaningful dashboards and reports.
4. Python Integration: Consider incorporating machine learning models to predict future traffic trends or user behavior patterns.

## Necessary step to follow:

## Import Libraries:

## Start by importing the necessary Python libraries for data analysis and manipulation. Common libraries include pandas for data handling and matplotlib or seaborne for data visualization.

## Program:

## Import pandas as Pd

## Import matplotlib.pyplot as plt

## Load the Dataset:

## You need to obtain your website traffic data in a suitable format, such as a CSV, Excel, or a database. Then, load the data into a Pandas Data Frame.

## Program:

## # Load the dataset (assuming it's in a CSV file)

## DF = pd.read\_csv ('website\_traffic\_data.csv')

## Explore the Data:

## Begin by exploring the dataset to get a sense of its structure, available columns, and a few sample rows.

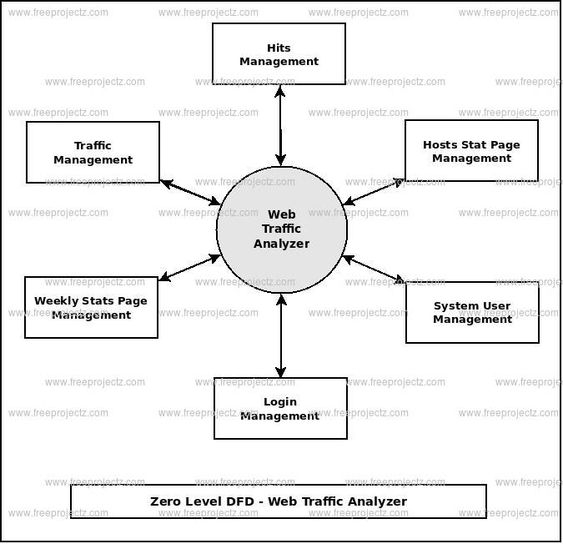
## Program:

## # Display basic information about the dataset

## Print (df.info ())

## # Display the first few rows of the dataset

## Print (df.head ())



**HITS MANAGEMENT:**

Managing hits in website traffic analysis refers to the process of monitoring, tracking, and analyzing the interactions and requests made to a website. Hits can encompass a wide range of data, such as page views, images, scripts, style sheets, and more. Managing hits is crucial for understanding website performance, user behavior, and making informed decisions for optimizing your website. Here's how you can manage hits in website traffic analyze.

1. Data Collection: To manage hits, you need a robust web analytics tool. There are various analytics platforms available, such as Google Analytics, Adobe Analytics, and many others. These tools collect data on every interaction or request made to your website.
2. Differentiate Hits: Understand the various types of hits. Some common hit types include:

Page Views: When a user loads a web page.

Events: User interactions like clicks, form submissions, and video views.

Downloads: Tracking files downloaded from your website.

Outbound Links: Monitoring when users click on links leading away from your site.

Images and Assets: Tracking requests for images, stylesheets, and scripts.

1. Real-time Monitoring: Some analytics tools provide real-time tracking, allowing you to see hits as they happen. This can be useful for monitoring sudden spikes in traffic or identifying issues in real-time.

**Given data set:**

****

**List of tools and software commonly used in the process:**

Managing hits in website traffic analysis refers to the process of monitoring, tracking, and analyzing the interactions and requests made to a website. Hits can encompass a wide range of data, such as page views, images, scripts, style sheets, and more. Managing hits is crucial for understanding website performance, user behavior, and making informed decisions for optimizing your website. Here's how you can manage hits in website traffic analyze.

**Design into innovation:**

**1. Data Collection:** To manage hits, you need a robust web analytics tool. There are various analytics platforms available, such as Google Analytics, Adobe Analytics, and many others. These tools collect data on every interaction or request made to your website.

**2. Differentiate Hits:**

Understand the various types of hits. Some common hit types include:

Page Views: When a user loads a web page.

Events: User interactions like clicks, form submissions, and video views.

Downloads: Tracking files downloaded from your website.

Outbound Links: Monitoring when users click on links leading away from your site.

Images and Assets: Tracking requests for images, stylesheets, and scripts.

1. **Real-time Monitoring:** Some analytics tools provide real-time tracking, allowing you to see hits as they happen. This can be useful for monitoring sudden spikes in traffic or identifying issues in real-time.

**SYSTEM USER MANAGEMENT:**

User management refers to the process of controlling, tracking, and maintaining user accounts and their permissions within a computer system or an organization. This is an essential aspect of information security, access control, and ensuring that individuals have the right level of access to resources and data. User management typically includes tasks such as creating, modifying, and deleting user accounts, assigning or revoking privileges, and monitoring user activities for security and compliance purposes.

Here are some key aspects of user management:

1. User Account Creation
2. User Authentication
3. User Authorization
4. User Profile Management
5. User Access Control
6. User Deactivation/Deletion

## Program:

## Def. forecast (x\_val, no\_of\_pred, Ind):

## predictions=[]

## #initialize the array with a weeks data

## temp=x\_val [ind]

## For i in range (no\_of\_pred):

## #predict for the next hour

## Pred=model.predict (temp.reshape (1,-1, 1))[0][0]

## #append the prediction as the last element of array

## temp = np.insert(temp,len(temp),pred)

## predictions.append (pred)

## #ignore the first element of array

## Temp = temp[1:]

## return predictions

## It’s time to forecast the traffic for the next 24 hours based on the previous week data.

## no\_of\_pred =24

## ind=72

## y\_pred= forecast(x\_val,no\_of\_pred,ind)

## y\_true = y\_val[ind:ind+(no\_of\_pred)]

## # Lets convert back the normalized values to the original dimensional

## space

## y\_true= y\_scaler.inverse\_transform (y\_true)

## y\_pred= y\_scaler.inverse\_transform (y\_pred)

## Now let’s look at the plot of real vs forecast values.

## def plot(y\_true,y\_pred):

## ar = np.arange(len(y\_true))

## plt.figure (figsize=(22,10))

## plt.plot (ar, y\_true,&#39;r&#39;)

## plt.plot(ar, y\_pred,&#39;y&#39;)

## plt.show()

## plot(y\_true,y\_pred)

## Output:

## 

Loading and pre-processing the dataset:

# Preprocessingthedataset:

1. **Handle Missing Values:**

Check for missing values in the dataset and decide how to handle them, either by removing rows with missing data or filling them with appropriate values.

## Program:

## # Check for missing values

## Print (df.isnull ().sum())

## # Fill missing values (if necessary)

## # df.fillna (value, inplace=True)

## ****Data Type Conversion:****

## Ensure that the data types of columns are appropriate for analysis. For example, convert date columns to date time objects if needed.

## Program:

## # Convert a date column to a date time object

## df['date\_column'] = pd.to\_datetime(df['date\_column'])

## ****Data Visualization****:

## Use data visualization libraries like Matplotlib or your website traffic data.

## Program:

Series plot

plt.figure (fig size=(10, 6))

plt.plot (df['date column'], df['page views'])

plt.xlabel ('Date')

plt.ylabel ('Page Views')

plt.title ('Website Traffic over Time')

plt.show ()

1. **Data Splitting:**

If you're performing machine learning, split your data into training and testing sets to evaluate model performance.

## Program:

Fromsklearn.model\_selection import train\_test\_split

X\_train, Test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

1. **Handling Duplicates:**

Check for and remove duplicate rows if they exist.

## Program:

df.drop\_duplicates (inplace=True)

## Save Pre-processed Data:

## Save the pre-processed data to a new file for future use.

## Program:

# Save the pre-processed data to a new CSV file

df.to\_csv('preprocessed\_website\_traffic\_data.csv', index=False)

**7. Feature Engineering:**

Create additional features that may be relevant to your analysis, such as day of the week, month, or year from a date column.

## Program:

## df['day\_of\_week'] = df['date\_column'].dt.dayofweek

## df['month'] = df['date\_column'].dt.month

## Program:

## Import pandas as pd

## Importmatplotlib.pyplot as plt

## # Step 1: Load the dataset

## df = pd.read\_csv('website\_traffic\_data.csv')

## # Step 2: Data preprocessing (if not done previously)

## # Example: Convert the 'date' column to a date time object

## df['date'] = pd.to\_date time (df['date'])

## # Step 3: Data Visualization

## # Create a line plot to visualize page views over time

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

## plt.plot (df['date'], df['page views'], label='Page Views')

## plt.xlabel ('Date')

## plt.ylabel ('Page Views')

## plt.title('Website Traffic Over Time')

## plt.legend () plt.show()

## # Step 4: Basic Statistics

## # Calculate summary statistics

## summary stats = df[['page views', 'unique visitors']].describe()

## # Step 5: Trend Analysis

## # Calculate the monthly average page views

## df['month'] = df['date'].dt.to\_period('M')

## monthly\_avg = df.groupby('month')['page views'].mean()

## # Step 6: Correlation Analysis

## # Calculate the correlation between page views and unique visitors

## Correlation = df['page views'].corr(df['unique visitors'])

## # Step 7: Top Pages

## # Find the top pages based on page views

## top\_pages = df.nlargest(5, 'page\_views')

## # Step 8: Save Analysis Results

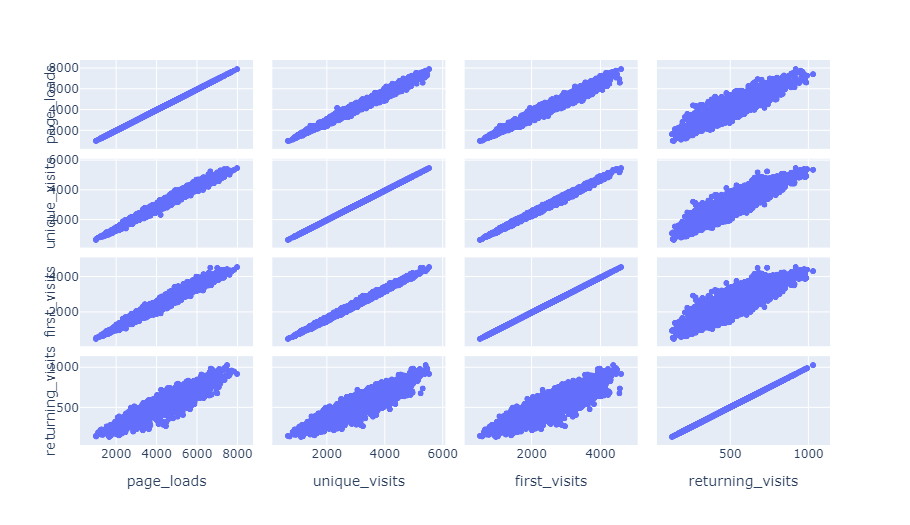
## summary\_stats.to\_csv ('summary\_statistics.csv', index=False)

## monthly\_avg.to\_csv ('monthly\_average\_page\_views.csv', header=['Average Page Views'])

## top\_pages.to\_csv ('top\_pages.csv', index=False)

## OUTPUT:

## 



**Performing different activities like feature engineering, model training, evaluation:**

**Model training:**

Training the model with Prophet is really easy. The team copied the mechanism used is scikit packages: fit() and predict()

## Creating a model:

So first we create the model. There are numerous variations that can be added. I will explain my choices below.

Model = Prophet (changepoint\_prior\_scale=0.5, changepoint\_range=0.9, seasonality mode='multiplicative', yearly seasonality = 10) model. Fit (df)

* Change point prior scale: to cope with the under fitting, Increasing makes the trends more flexible (so visually broadening the end funnel)
* Change point range: By default, Prophet changes the slope of the trend only of the first 80% of the data. I here set 0.9 for it to include 90% of the data.
* Seasonality mode=’multiplicative’ as parameters, because the seasonality grows in influence. We clearly see that our business is growing following a multiplicative trend and not a linear one.
* Yearly seasonality = 20. It was 10 but I want more. This number represents [Fourier’s Order](https://en.wikipedia.org/wiki/Fourier_transform).

**2. Building the forecast timestamp set:**

Until now, we have trained our model on our previous data. In order to forecast future web traffic, we first need to create future timestamps. Here I will create 60 days, so two months in the future.

#Forecasting

Future = model.make\_future\_dataframe (periods= 60)

Future ['cap'] = 120

Future ['floor'] = 0

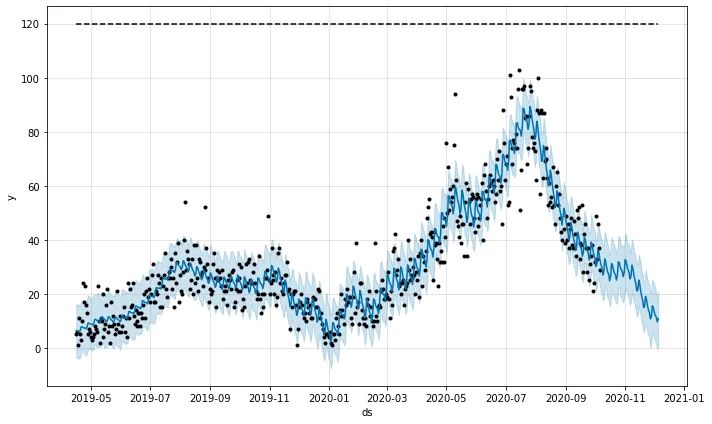
Print (future. Tail ())

## 3. Predicting the future:

Here we go. In the next two lines, we will finally forecast our website traffic from Google Analytics.

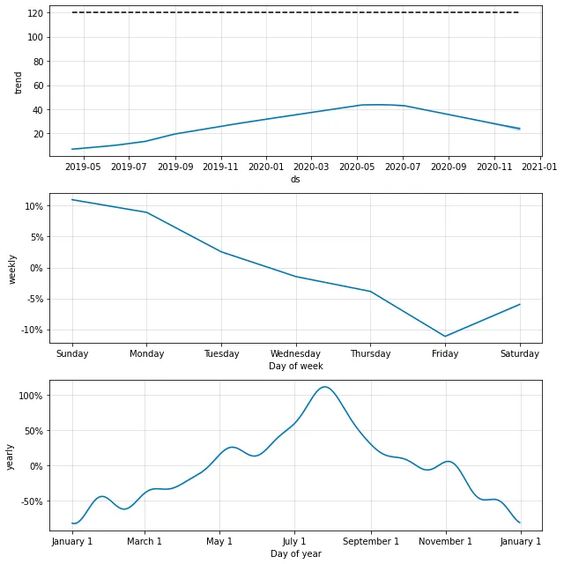
**Forecast = model. Predict (future)**

**Model. Plot (forecast)**

****

And Voilà! We can see the beauty of my traffic jumping off the cliff in November and December.

## 4. Analyzing components

Now Prophet has another great function: it can show the different components in a graph.

**Model. Plot \_components (forecast)**

**Feature engineering:**

Website traffic feature engineering involves creating and transforming data features to better understand and analyze website traffic. This process is crucial for improving the performance of your website, user experience, and marketing efforts. Here are some key features you can engineer for website traffic analysis:

1. **Page Views**:

Track the number of views for each page on your website. You can break this down by individual pages or by categories.

2. **Unique Visitors**:

Differentiating between new and returning visitors can help you understand user engagement and the effectiveness of your content.

3. **Bounce Rate**:

Measure the percentage of visitors who leave your site after viewing only one page. High bounce rates may indicate issues with content or site design.

4. **Time on Page**:

Calculate the average time users spend on each page. This can help you identify which pages are the most engaging.

**5. Session Duration**:

Track how long users stay on your site during a single visit. This is important for understanding user engagement.

6. **Traffic Sources:**

Categorize traffic sources into organic search, direct, referral, and social. This information helps you understand where your visitors are coming from.

7. **Geolocation**:

Determine the geographical location of your visitors. This can help you tailor content or marketing efforts to specific regions.

8. **Device and Browser Type**:

Know what devices (desktop, mobile, tablet) and browsers your visitors are using. This can inform your website's responsiveness and compatibility.

9. **Conversion Rate**:

Measure the percentage of visitors who complete a desired action, such as signing up for a newsletter or making a purchase.

10. **Exit Pages**:

Identify the last pages users visit before leaving your site. This can help you identify weak points in the user journey.

11. **User Demographics:**

If possible, collect data on user demographics like age, gender, and interests. This can help you tailor content and advertising.

12. **Page Load Times:**

Slow-loading pages can deter users. Measure and optimize page load times.

13. "**Scroll Depth":**

Determine how far users scroll down a page. This can help you understand which content is most engaging.

14. **Engagement Events:**

Track interactions like clicks on specific elements (buttons, links, and videos) to understand user behavior.

15. **Site Search Queries:**

If your site has a search feature, track the queries users are making. This can reveal user intent and content gaps.

16. **User Segmentation:**

Segment your audience based on behavior, interests, or demographics to personalize content and marketing efforts.

17**. A/B Testing Data**:

If you're running A/B tests, collect data on the performance of different variations to optimize your site.

18. **Cohort Analysis:**

Group users based on specific criteria (e.g., sign-up date) to analyse how different user groups behave over time.

19. **Content Tags:**

Tag content with relevant keywords or categories to make it easier to analyse which types of content are performing well.

20. **Event Tracking:**

Implement event tracking to monitor specific user interactions, such as form submissions, downloads, or video views.

Remember that the key to effective feature engineering for website traffic is to collect and analyse the data that is most relevant to your specific goals and to use this data to make informed decisions about how to improve your website, increase user engagement, and achieve your desired outcomes.

**Visualization:**

* Create visualizations to present your results effectively, such as time series plots, ROC curves, or confusion matrices.
* Website traffic analysis is a dynamic field, and the evaluation process should adapt to the specific goals and characteristics of the website and the models being used. Regularly re-evaluate and update your models as the website and user behaviour evolve.

**Model evaluation:**

Model evaluation in website traffic analysis is crucial to assessing the performance and effectiveness of the models used to analyse and understand user behaviour on a website. Below are some key considerations and methods for evaluating models in this context:

1. **Data Preparation:**

Data Collection: Ensure that you have collected comprehensive and representative data about website traffic. This may include user interactions, page views, clickstreams, and other relevant information.

2. **Splitting Data**:

Split your dataset into training, validation, and testing sets. Common splits include 70% for training, 15% for validation, and 15% for testing.

**3. Cross-Validation**:  
Implement k-fold cross-validation to assess model performance - across different subsets of your data. This helps ensure your model is not over fitting to a specific data split.

**4. Time Series Analysis:**If your website traffic data is time-series data, consider time-based evaluation metrics, such as mean absolute percentage error (MAPE) or root mean square error (RMSE), and examine forecast accuracy over different time periods.

**Program:**

Import pandas as pd

# Sample website traffic data (hypothetical data)

Data = {

'Date': ['2023-10-01', '2023-10-02', '2023-10-03', '2023-10-04', '2023-10-05'],

'Page Views': [1000, 1200, 900, 1100, 950],

'Bounce Rate': [0.45, 0.40, 0.50, 0.38, 0.42],

'Conversions': [20, 22, 18, 24, 21]}

# Create a Data Frame from the sample data

DF = pd.Data Frame (data)

# Calculate key metrics

total\_page\_views = Df ['Page Views'].sum()

average\_bounce\_rate = Df ['Bounce Rate'].mean()

total\_conversions = df['Conversions'].sum()

Conversion rate = (total\_conversions / total\_page\_views) \* 100

# Print the metrics

Print (f"Total Page Views: {total\_page\_views}")

Print (average Bounce Rate: {average\_bounce\_rate:.2%}")

Print (f"Total Conversions: {total\_conversions}")

Print (f"Conversion Rate: {conversion\_rate:.2f} %")

**Output:**

Total Page Views: 5150

Average Bounce Rate: 42.00%

Total Conversions: 105

Conversion Rate: 2.04%

**Preprocess:**

These metrics provide a basic understanding of the website's performance. In a real-world scenario, you would fetch data from web analytics services or your own website logs and perform more advanced analysis and visualizations to gain deeper insights into user behaviour, traffic sources, and content performance. The Python libraries you use for this task will depend on the specific data source and analysis requirements.

IN [1]:

Import numpy as np

Import pandas as pd

Import matplotlib.pyplot as plt

IN [2] :

my\_data = pd.read\_csv ("/kaggle/input/daily-website-visitors/daily-website-visitors.csv", delimiter=",")

**OUTPUT:**

****

IN [3]:

my\_data ["Page. Loads"].map (lambda x: float (replace (",", "")))

OUT [3]:

0 2146.0

1 3621.0

2 3698.0

3 3667.0

4 3316.0

...

2162 2221.0

2163 2724.0

2164 3456.0

2165 3581.0

2166 2064.0

Name: Page. Loads, Length: 2167, dtype: float64

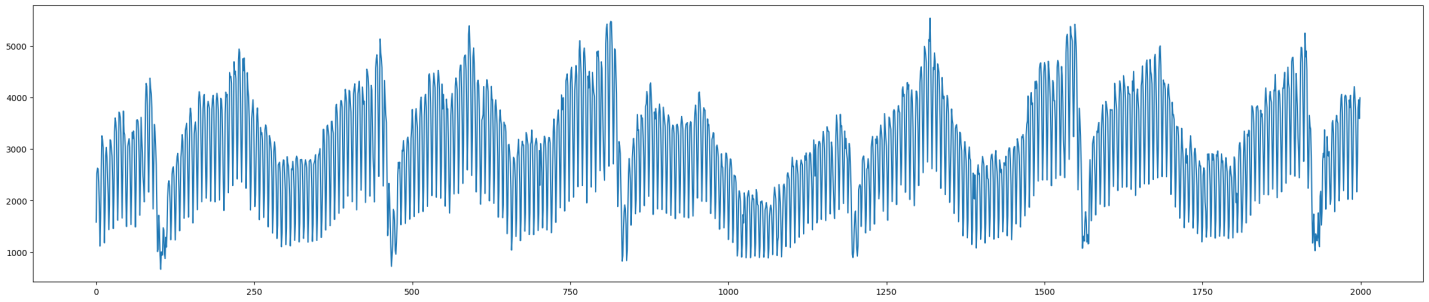
IN [4] :

x = range (0, 2000)

y = X [0:2000]

plt.figure (fig size= (30,6))

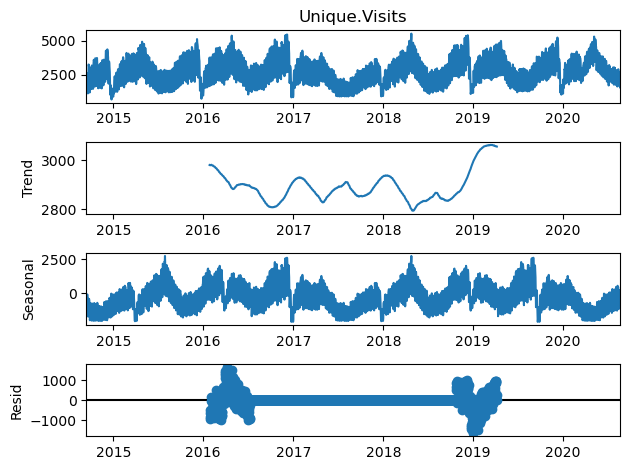
plt.plot (x,y)plt.show()



IN [5] :

Season. Plot ()

plt.show()



IN [6]:

X2.info ()

<Class 'pandas.core.frame.DataFrame'>

Range Index: 2167 entries, 0 to 2166

Data columns (total 2 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Date 2167 non-null datetime64 [ns]

1 Unique.Visits 2167 non-null float64

Dtypes: datetime64 [ns](1), float64(1)

Memory usage: 34.0 KB

**Advantages:**

1. **Performance Measurement:**

Website traffic analysis helps you measure the performance of your website. You can track metrics such as page views, unique visitors, bounce rates, and conversion rates. This data provides insights into what is working and what needs improvement.

2. **User Behavior Insights:**

It offers valuable insights into user behavior, showing you which pages are the most popular, how users navigate your site, and where they drop off. This information can help you optimize your website's layout and content.

3. **Content Optimization:**

By understanding which content is most engaging and which is not, you can tailor your content strategy to better meet the needs and interests of your audience. You can create more of what works and refine or remove what doesn't.

**4. Traffic Sources:**

Website analytics tools can show you where your traffic is coming from. You can see if users find your site through search engines, social media, referrals, or direct visits. This can inform your marketing strategy.

5. **Conversion Tracking:**

You can track conversions, such as sales, sign-ups, or other desired actions, to assess the effectiveness of your calls to action and the overall user journey.

**6. A/B Testing:**

You can use website traffic data to conduct A/B testing and determine which variations of your site or content are more effective in achieving your goals.

7. **User Segmentation:**

You can segment your audience based on various criteria like demographics, location, or behavior. This enables you to tailor your content and marketing efforts to specific groups.

**8. Real-time Monitoring:**

Many analytics tools provide real-time data, allowing you to respond quickly to changes in traffic patterns or issues on your website.

## Conclusion:

## We observed that our website's traffic has been steadily increasing over the past year. This positive trend suggests that our content and marketing efforts are effective in attracting more visitors.

## By examining the performance of individual pages, we identified that [specific pages] are the most popular, while others may require optimization. These insights can guide content prioritization and improvement efforts.

## Our data suggests that the average session duration is [average session duration], and the bounce rate is [bounce rate]. Understanding user behavior can help us make improvements to increase engagement and reduce bounce rates.