Website traffic analysis :

Website traffic analysis is the process of examining the data related to visitors and their interactions with a website. It provides valuable insights into how users engage with your site, helping you make data-driven decisions to improve its performance and user experience. Key aspects of website traffic analysis include tracking the number of visitors, their demographics, referral sources, popular pages, and user behavior. This data is typically collected using web analytics tools like Google Analytics. By analyzing this information, website owners can optimize content, design, and marketing strategies to enhance user engagement, conversion rates, and overall site effectiveness.

import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O*

*# Input data files are available in the read-only*

import os

for dirname, \_, filenames **in** os.walk('/kaggle/input'):

for filename **in** filenames:

print(os.path.join(dirname, filename))

/kaggle/input/daily-website-visitors/daily-website-visitors.csv

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats import mode

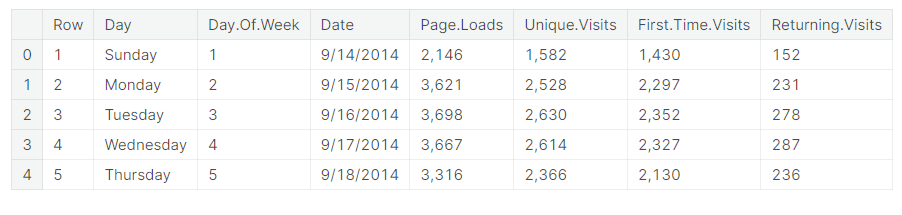
* *Installing Facebook Prophet*
* *Loading the data*

data\_path = "../input/daily-website-visitors/daily-website-visitors.csv"

data = pd.read\_csv(data\_path)

linkcode

data.head()



*# Function to remove commas*

def remove\_commas(x):

return float(x.replace(',', ''))

*# Apply the preprocessing functions*

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

data['Page.Loads'] = data['Page.Loads'].apply(lambda x : remove\_commas(x))

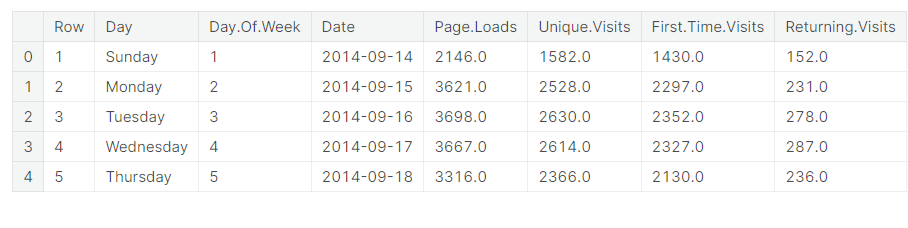
data['Unique.Visits'] = data['Unique.Visits'].apply(lambda x : remove\_commas(x))

data['First.Time.Visits'] = data['First.Time.Visits'].apply(lambda x : remove\_commas(x))

data['Returning.Visits'] = data['Returning.Visits'].apply(lambda x : remove\_commas(x))

linkcode

data.head()



*# Data preprocessing*

*'''*

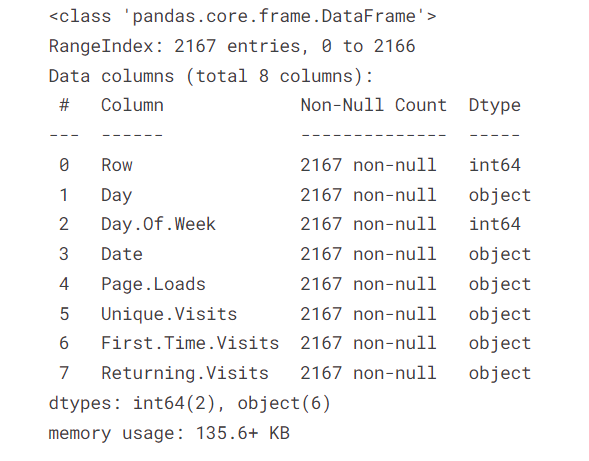
*1.Convert Date into Datetime format.*

*2.Removing ',' from Page.Loads, Unique.Visits, First.Time.Visits, Returning.Visits.*

*3.Convert the above values into float.*

*'''*

data.info()



Exploratory Data Analysis

*# Frequency distribution of each continuous column*

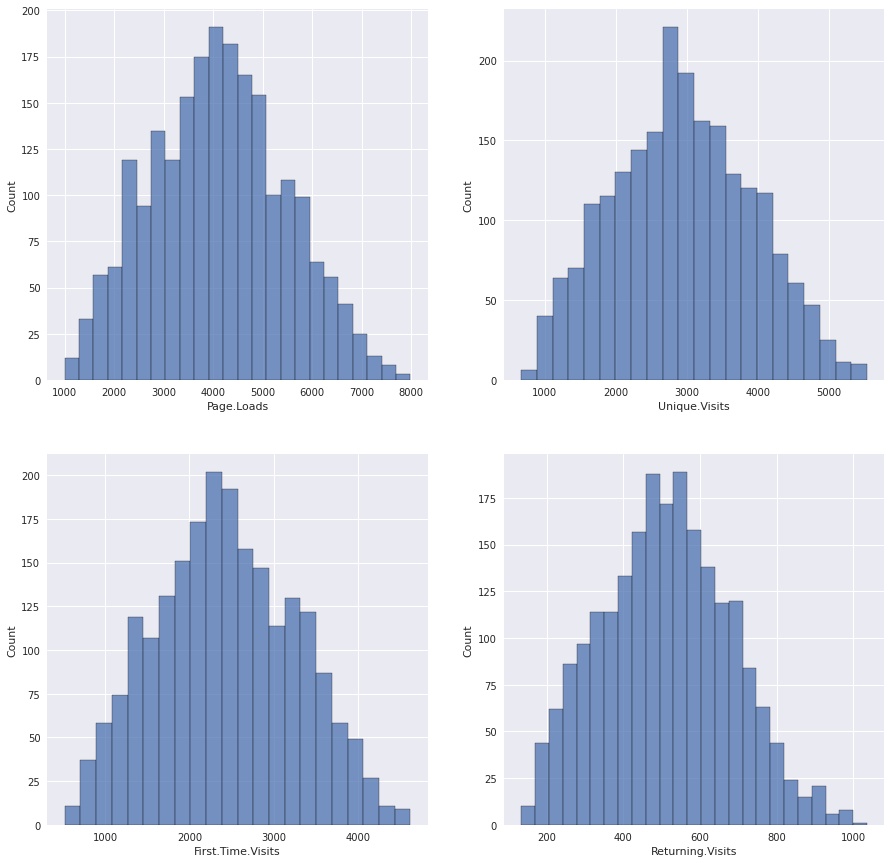
cols\_to\_plot = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']

plt.figure(figsize=(15, 15))

for i, col **in** enumerate(cols\_to\_plot):

plt.subplot(2, 2, i+1)

sns.histplot(data=data, x=col)



**Visually the distributions appear to be normal**

def check\_normality(data, col):

*# Compute mean*

mean = int(np.mean(data[col]))

median = int(np.median(data[col]))

mode\_ = int(mode(data[col])[0][0])

print("mean", ":", mean, "median", ":", median, "mode", ":", mode\_)

if mean == median == mode\_:

print("**{}** Distribution is Normal".format(col))

elif mean > median **and** mean > mode\_ **and** mode\_ < median:

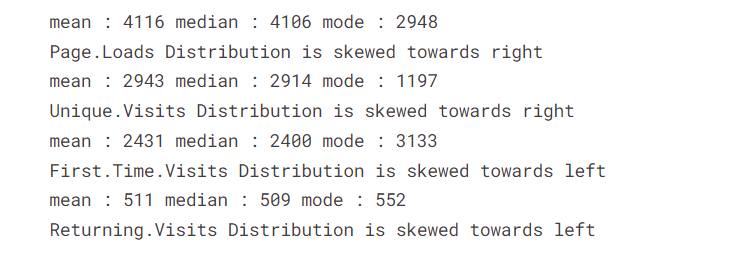
print("**{}** Distribution is skewed towards right".format(col))

else:

print("**{}** Distribution is skewed towards left".format(col))

for col **in** cols\_to\_plot:

check\_normality(data, col)



**Although, the distributions seem to look normal, but they fail the normality test. Hence, it can be concluded that the distributions are not normal.**

## **Day wise analysis**

def day\_wise\_EDA(day):

sun\_data = day\_grouped\_data.get\_group(day)

figure, ax = plt.subplots(2, 2, figsize=(17, 15))

plt.style.use('seaborn')

ax1 = ax[0]

ax2 = ax[1]

*# Plot the Number of Page Loads with time*

print("==================================================================**{}** ANALYSIS======================================================".format(day.upper()))

ax1[0].plot(sun\_data['Date'], sun\_data['Page.Loads'])

ax1[0].set\_xlabel("Date")

ax1[0].set\_ylabel("Number of Page Loads")

*# Plot the Number of Unique Visits with time*

ax1[1].plot(sun\_data['Date'], sun\_data['Unique.Visits'])

ax1[1].set\_xlabel("Date")

ax1[1].set\_ylabel("Number of Unique Visits")

*# Plot the Number of First Time visits with time*

ax2[0].plot(sun\_data['Date'], sun\_data['First.Time.Visits'])

ax2[0].set\_xlabel("Date")

ax2[0].set\_ylabel("Number of First Time visits")

*# Plot the Number of Returning visits with time*

ax2[1].plot(sun\_data['Date'], sun\_data['Returning.Visits'])

ax2[1].set\_xlabel("Date")

ax2[1].set\_ylabel("Number of Returning visits")

figure.show()

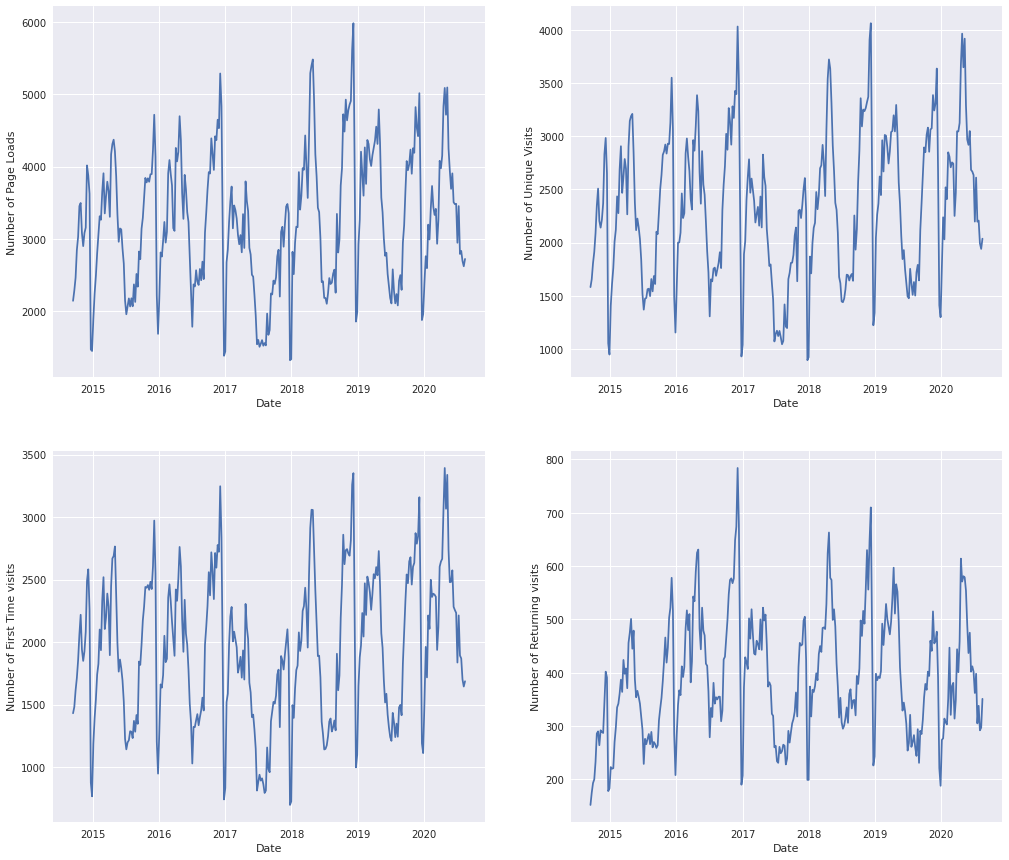
linkcode

*# Call the above function for every day*

*# 1. Sunday*

day\_wise\_EDA('Sunday')

==========================SUNDAY ANALYSIS============================



*BAR GRAPH :*

*# Plot the Bargraph for every continuous variable across day*

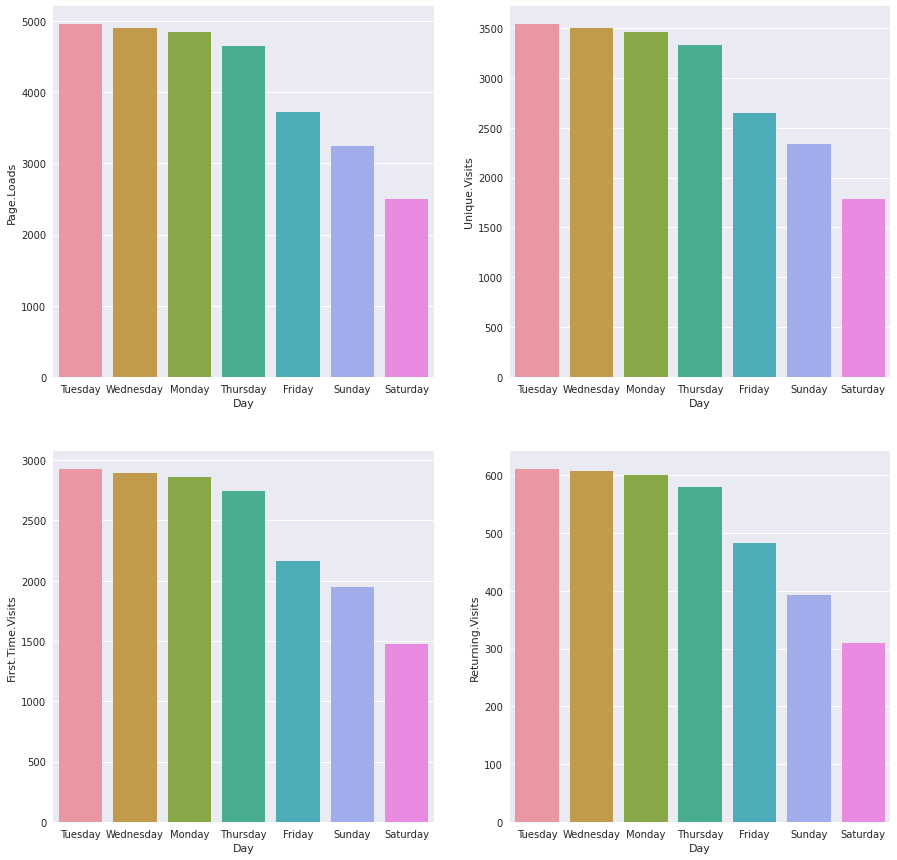
cols\_to\_plot = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']

plt.figure(figsize=(15, 15))

for i, col **in** enumerate(cols\_to\_plot):

plt.subplot(2, 2, i+1)

sns.barplot(data=avg\_day\_data.sort\_values(by=col, ascending=False), x='Day', y=col)



*CORRELATION HEATMAP :*

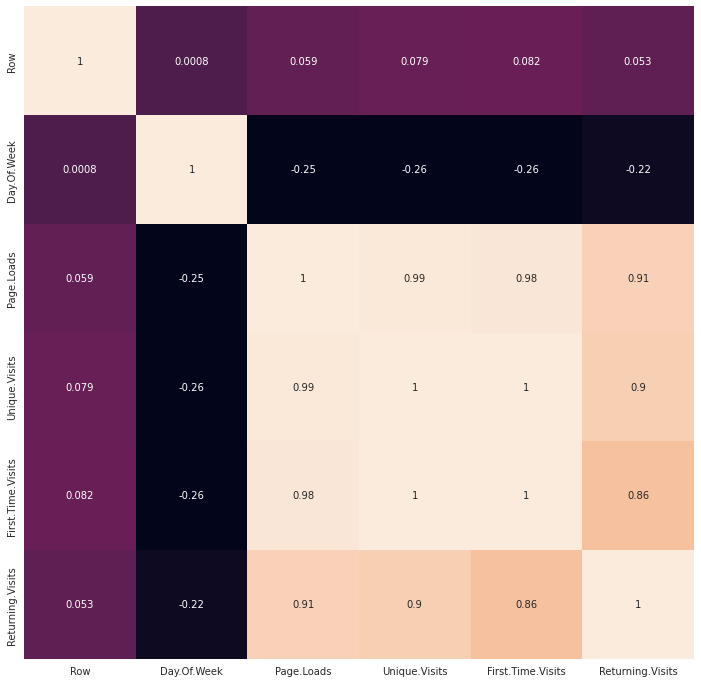
*#Plot the correlation heatmap*

corr\_matrix = data.corr()

plt.figure(figsize=(12,12))

sns.heatmap(corr\_matrix, annot=True, cbar=False)

plt.show()



Conclusion :

Incorporating chatbots and NLP into your website traffic analysis strategy can provide a more interactive and insightful way to understand user behavior and enhance the user experience. These tools can offer real-time data and actionable insights that help you make data-driven decisions to improve your website's performance.