Web Traffic Analysis

Analyzing data related to the behavior of users on a website or web application

Web traffic analytics refers to collecting data about who comes to your website and what they do when they get there. That data is crucial to building effective sales and marketing strategies

Identifying the most popular content on a website

This project can be used to identify the pages, blog posts, or other types of content that are most popular with visitors. This information can be used to improve the website's content strategy and attract more traffic.

Analyzing the behavior of different visitor segments

- This project can be used to analyze the behavior of different visitor segments, such as new visitors, returning visitors, and customers. This information can be used to improve the website's user experience and conversion rate.

Tracking the performance of marketing campaigns

- This project can be used to track the performance of different marketing campaigns, such as search engine marketing, social media marketing, and email marketing. This information can be used to identify which campaigns are most effective at driving traffic and conversions.

Analyzing the impact of website changes on traffic

- This project can be used to analyze the impact of website changes, such as new features, design changes, and content changes, on traffic. This information can be used to make informed decisions about future website development and improvement.

Predicting future website traffic

- This project can be used to develop machine learning models to predict future website traffic. This information can be used to plan for future resource needs and marketing campaigns

- Choose a web traffic analysis tool. Consider your specific needs and requirements when choosing a tool

For example:

- if you need to track competitor traffic, you may want to choose a tool that offers competitive analysis features.
- Set up the web traffic analysis tool on your website. This typically involves adding a tracking code to your website's header or footer.
- Once the tracking code is installed, the tool will start collecting data about your website traffic.

- Collect data over a period of time. The more data you collect, the more accurate and insightful your analysis will be.

- It is recommended to collect data for at least a few months before starting our analysis.

- Analyze the data using a data analytics tool. Once you have extracted the data, you can use a data analytics tool to analyze it and gain insights into our website traffic and visitor behavior.

Here is our website traffic coming from sourse Is it from organic search, social media, paid advertising,or other sources

our website visitors if is their age, gender, location, and interests we have to extract

How many pages are visitors viewing on our website How long are they staying on your website

What percentage of visitors are leaving our website after viewing only one page

Before we start analyzing our data, it is important to clean it to remove any errors or inconsistencies.

Segment our data by different criteria, such as traffic source, visitor demographics, and pageviews. This will help you to identify trends and patterns in our data.

Compare our data over time to see how your website traffic and visitor behavior is changing.

Visualization tools can help us to understand your data more easily and quickly. For example, you can use charts and graphs to visualize trends and patterns in our data

In Time series analysis we have to analysis statistical data that can be used to identify trends and patterns in historical data. This information can then be used to predict future values.

Machine learning algorithms can be trained on statical data to predict future values. There are a variety of different machine learning algorithms we can use various prediction, such as linear regression, support vector machines, and random forests.

The more data you have, the more accurate our predictions will be. Consider using data from a variety of sources, such as Google Analytics, social media, and email marketing.

Before you train a prediction model, it is important to clean our data to remove any errors or inconsistencies.

Choose a prediction method that is appropriate for our data and the insights you are trying to gain.

Once you have trained a prediction model, it is important to evaluate its performance on a held-out test set. This will help you to identify any areas where the model needs improvement.

As our website traffic changes, you will need to update our prediction model to reflect these changes.

This includes removing incomplete, inaccurate, irrelevant, corrupt, or incorrectly formatted data.

You can either remove rows with missing values or impute the missing values using a variety of methods, such as the mean, median, or mode.

This includes converting dates and times to a consistent format, and ensuring that all categorical data is encoded in a consistent way.

If you are using data from multiple sources, you will need to merge the data into a single dataset. This may involve resolving duplicate data and converting data to a consistent format.

You can use statistical methods to identify trends and patterns in our data. For example, you could identify the most popular pages on our website or the days of the week when you receive the most traffic.

You can segment your audience based on different criteria, such as traffic source, visitor demographics, and pageviews. This can help you to better understand our audience and target our marketing messages more effectively.

You can use web traffic analysis data to measure the effectiveness of your marketing campaigns. For example, you could track the number of visitors from different traffic sources or the number of conversions that are generated from each campaign.

Web Traffic Analysis

Monitoring web traffic requires information such as the total number of visitors, average page views per visitor, most popular pages, average visits by visitors.

Remove duplicate observations

Duplicate data most often occurs during the data collection process. This typically happens when you combine data from multiple places, or receive data from clients or multiple departments.

Filter unwanted outliers

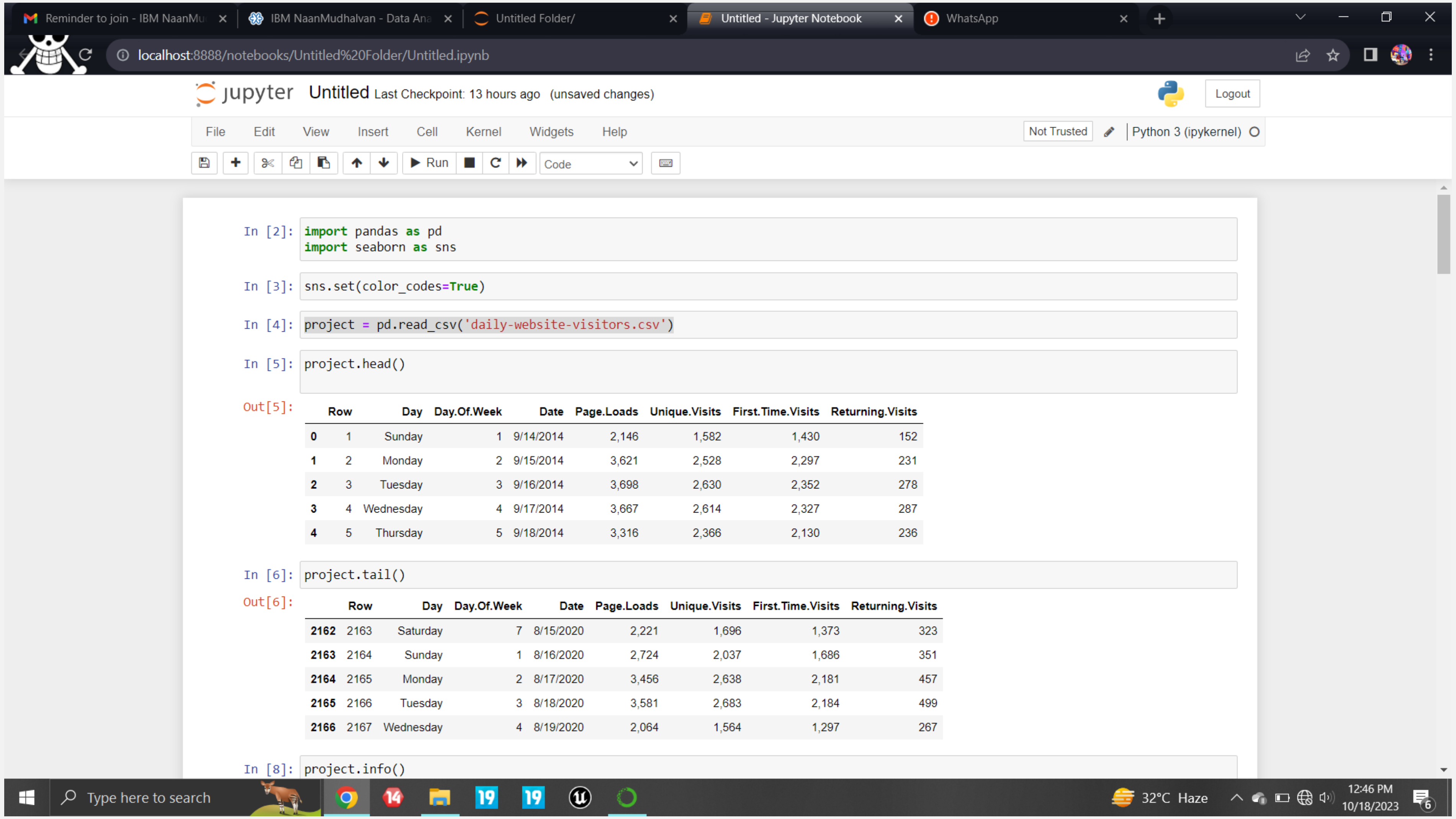
Outliers are unusual values in your dataset. They’re significantly different from other data point and can distort your analysis and violate assumptions. Removing them is a subjective practice and depends on what you’re trying to analyze.

Fix structural errors

Structural errors are things like strange naming conventions, typos, or incorrect capitalization. Anything that is inconsistent will create mislabeled categories.

The pandas have been installed into the system, we need to import the library

Seaborn is a library for making statistical graphics in Python


Reminder to join - IBM NaanMu
IBM NaanMudhalvan - Data Ana
Untitled Folder/
Untitled - Jupyter Notebook
WhatsApp
localhost:8888/notebooks/Untitled%20Folder/Untitled.ipynb
jupyter Untitled Last Checkpoint: 13 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Type here to search
32°C Haze
12:46 PM
10/18/2023

```
In [2]: import pandas as pd
        import seaborn as sns

In [3]: sns.set(color_codes=True)

In [4]: project = pd.read_csv('daily-website-visitors.csv')

In [5]: project.head()

Out[5]:
```

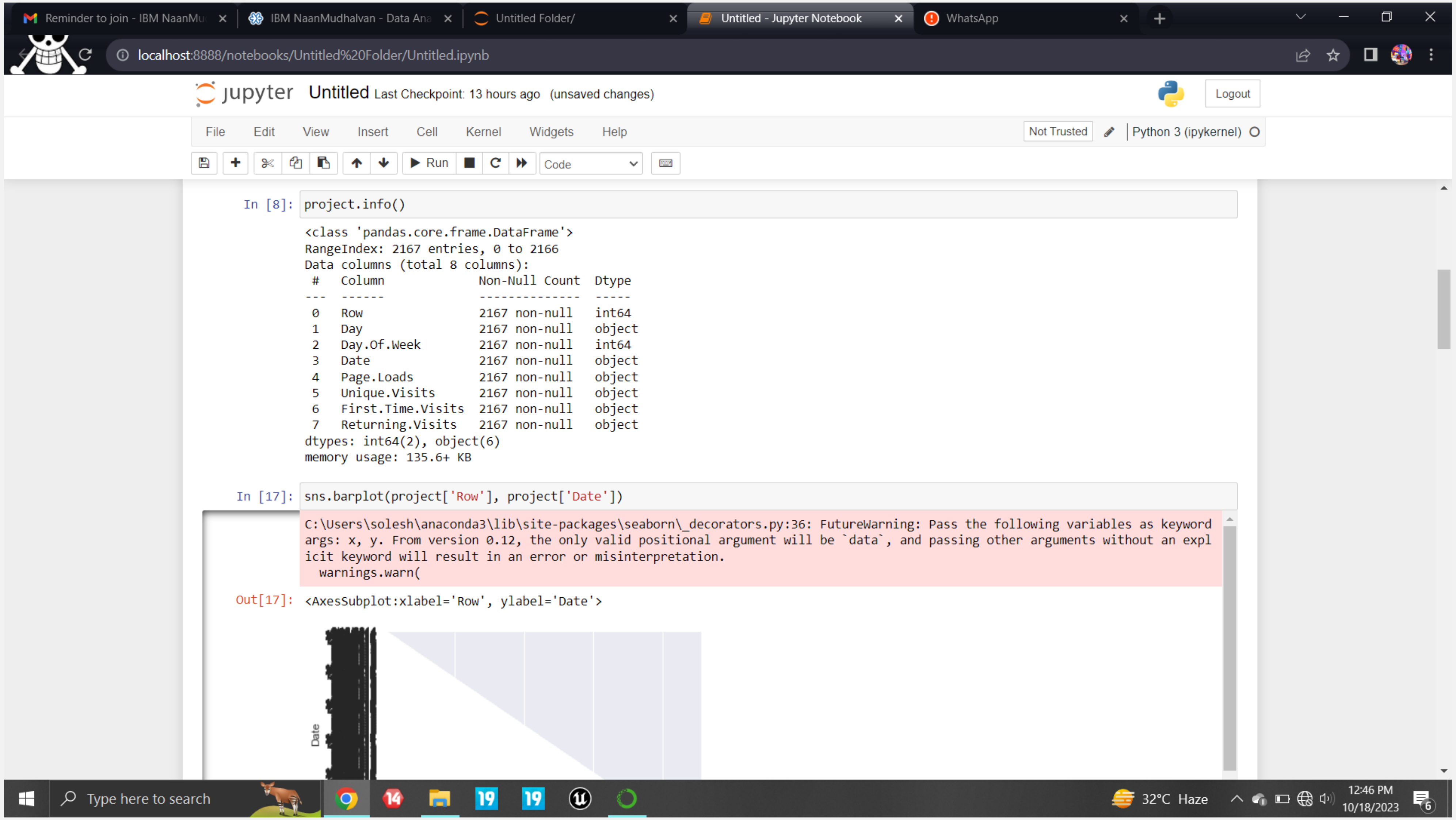
	Row	Day	Day.Of.Week	Date	Page.Loads	Unique.Visits	First.Time.Visits	Returning.Visits
0	1	Sunday	1	9/14/2014	2,146	1,582	1,430	152
1	2	Monday	2	9/15/2014	3,621	2,528	2,297	231
2	3	Tuesday	3	9/16/2014	3,698	2,630	2,352	278
3	4	Wednesday	4	9/17/2014	3,667	2,614	2,327	287
4	5	Thursday	5	9/18/2014	3,316	2,366	2,130	236

```
In [6]: project.tail()

Out[6]:
```

	Row	Day	Day.Of.Week	Date	Page.Loads	Unique.Visits	First.Time.Visits	Returning.Visits
2162	2163	Saturday	7	8/15/2020	2,221	1,696	1,373	323
2163	2164	Sunday	1	8/16/2020	2,724	2,037	1,686	351
2164	2165	Monday	2	8/17/2020	3,456	2,638	2,181	457
2165	2166	Tuesday	3	8/18/2020	3,581	2,683	2,184	499
2166	2167	Wednesday	4	8/19/2020	2,064	1,564	1,297	267

```
In [8]: project.info()
```

The information contains the number of columns, column labels, column data types, memory usage, range index, and the number of cells in each column



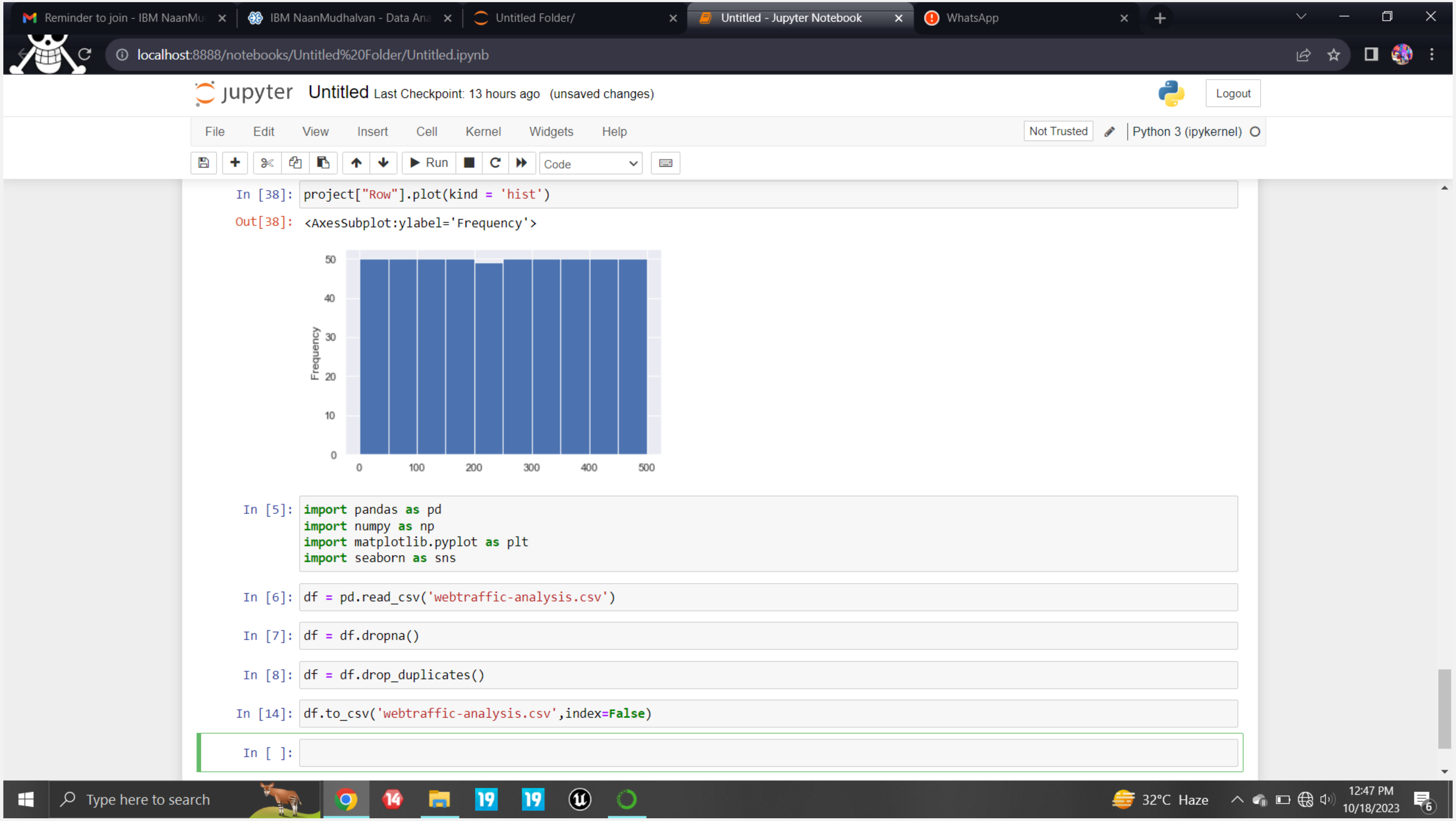
Reminder to join - IBM NaanMu
IBM NaanMudhalvan - Data Ana
Untitled Folder/
Untitled - Jupyter Notebook
WhatsApp
localhost:8888/notebooks/Untitled%20Folder/Untitled.ipynb
jupyter Untitled Last Checkpoint: 13 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Type here to search
32°C Haze
12:46 PM
10/18/2023

```
In [8]: project.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2167 entries, 0 to 2166
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Row                 2167 non-null   int64
 1   Day                 2167 non-null   object
 2   Day.Of.Week         2167 non-null   int64
 3   Date                2167 non-null   object
 4   Page.Loads          2167 non-null   object
 5   Unique.Visits       2167 non-null   object
 6   First.Time.Visits   2167 non-null   object
 7   Returning.Visits    2167 non-null   object
dtypes: int64(2), object(6)
memory usage: 135.6+ KB

In [17]: sns.barplot(project['Row'], project['Date'])

C:\Users\solesh\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword
args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an expl
icit keyword will result in an error or misinterpretation.
  warnings.warn(

Out[17]: <AxesSubplot:xlabel='Row', ylabel='Date'>
```

The dropna() method returns a new inplace parameter is set to True, in that case the dropna method does the removing in the original DataFrame instead.

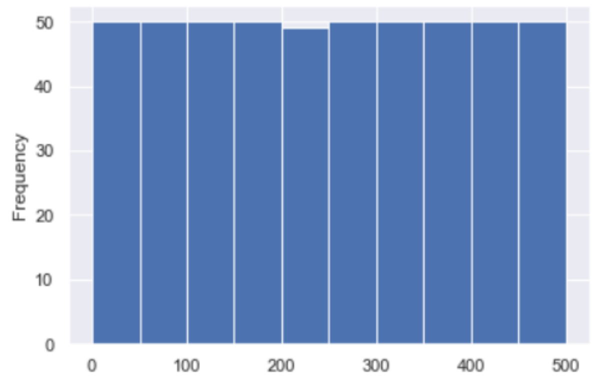
Use the subset parameter if only some specified columns should be considered when looking for duplicates.

After compliting the process Extract the cleaned data set then move to the next visualization



Reminder to join - IBM NaanMu
IBM NaanMudhalvan - Data Ana
Untitled Folder/
Untitled - Jupyter Notebook
WhatsApp
localhost:8888/notebooks/Untitled%20Folder/Untitled.ipynb
jupyter Untitled Last Checkpoint: 13 hours ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [38]: project["Row"].plot(kind = 'hist')

Out[38]: <AxesSubplot:ylabel='Frequency'>
```

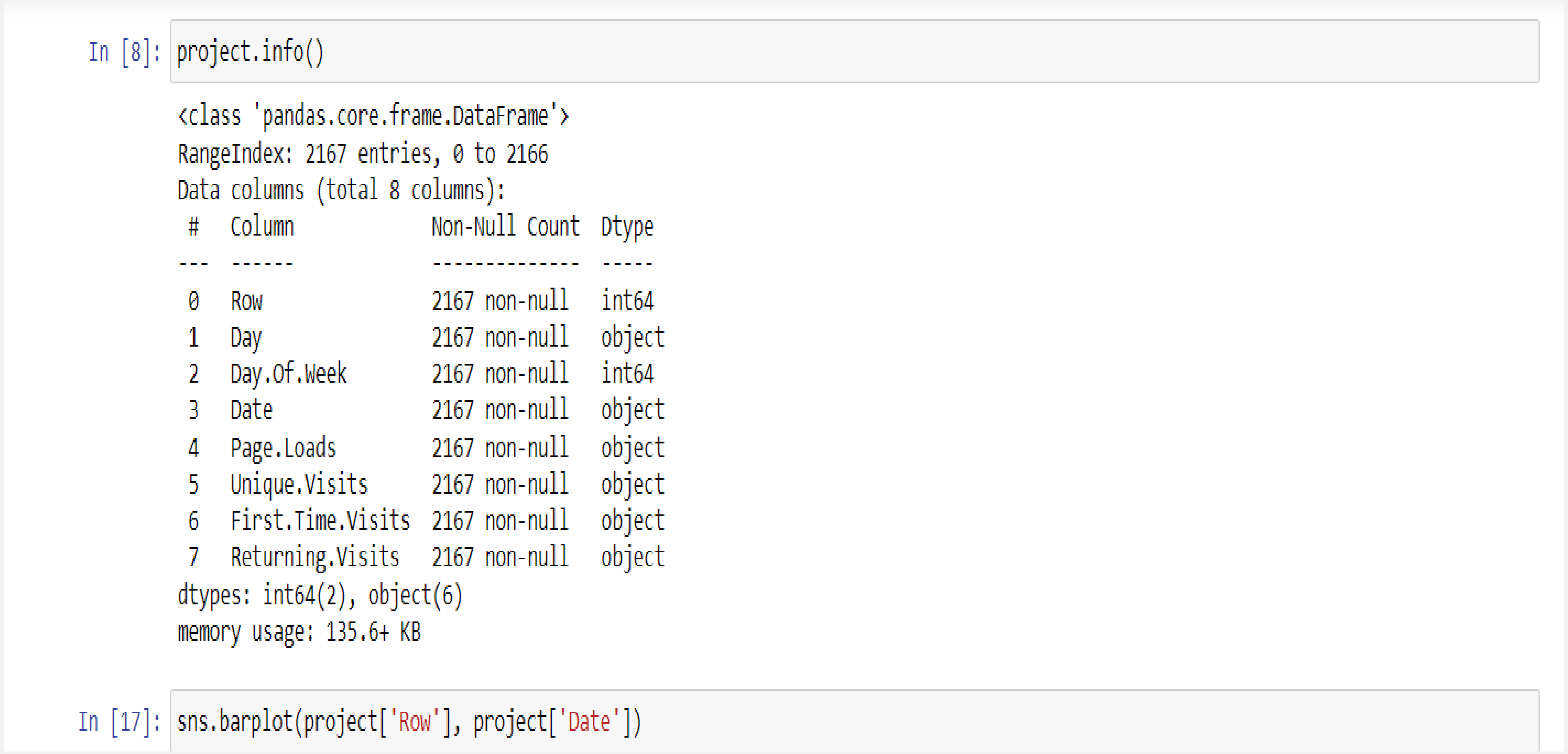


```
In [5]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns

In [6]: df = pd.read_csv('webtraffic-analysis.csv')

In [7]: df = df.dropna()

In [8]: df = df.drop_duplicates()

In [14]: df.to_csv('webtraffic-analysis.csv',index=False)

In [ ]:
```

WEB TRAFFIC ANALYTICS

Monitoring web traffic requires information such as the total number of visitors, average page views per visitor, most popular pages, average visits by visitors.

Duplicate data most often occurs during the data collection process. This typically happens when you combine data from multiple places, or receive data from clients or multiple departments.



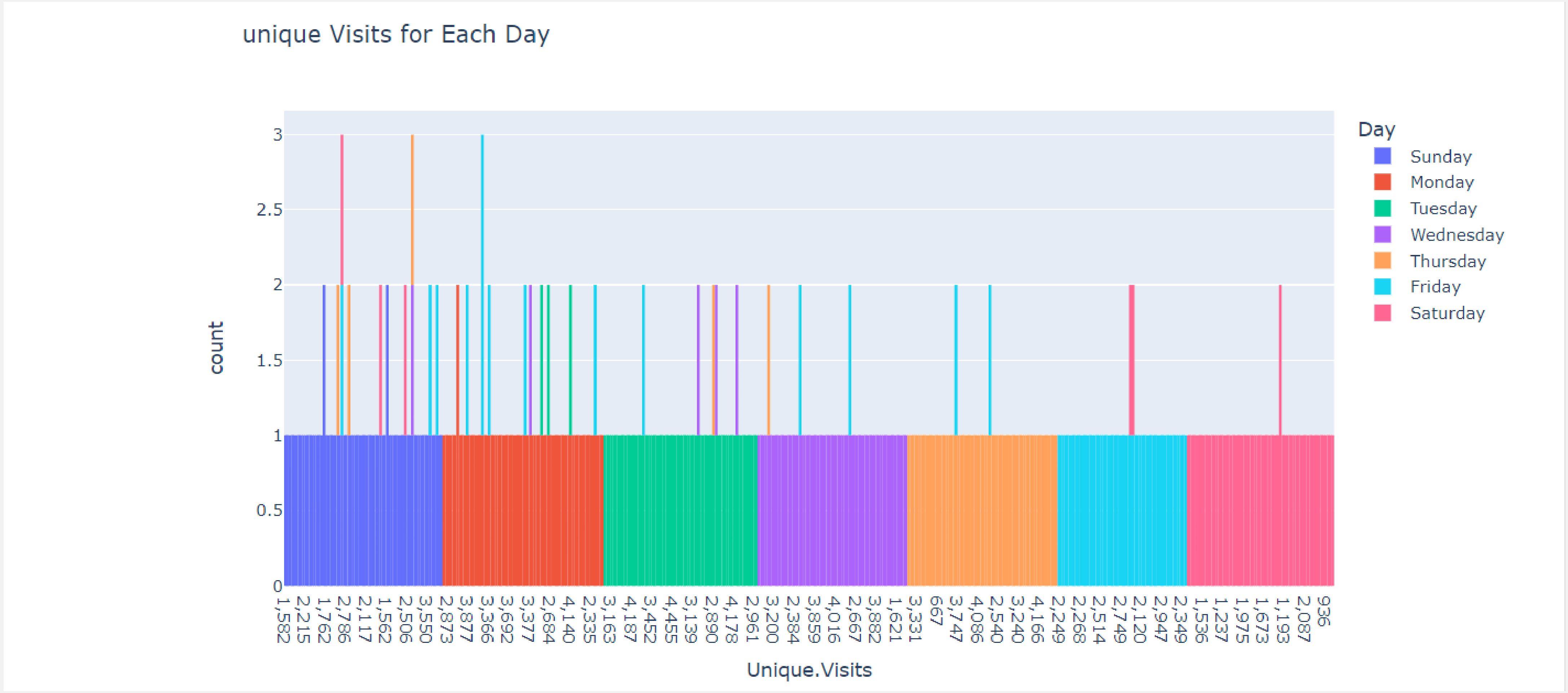
In [8]:
```
project.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2167 entries, 0 to 2166
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Row                 2167 non-null   int64 
 1   Day                 2167 non-null   object
 2   Day.Of.Week         2167 non-null   int64 
 3   Date                2167 non-null   object
 4   Page.Loads          2167 non-null   object
 5   Unique.Visits       2167 non-null   object
 6   First.Time.Visits   2167 non-null   object
 7   Returning.Visits    2167 non-null   object
dtypes: int64(2), object(6)
memory usage: 135.6+ KB
```

In [17]:
```
sns.barplot(project['Row'], project['Date'])
```

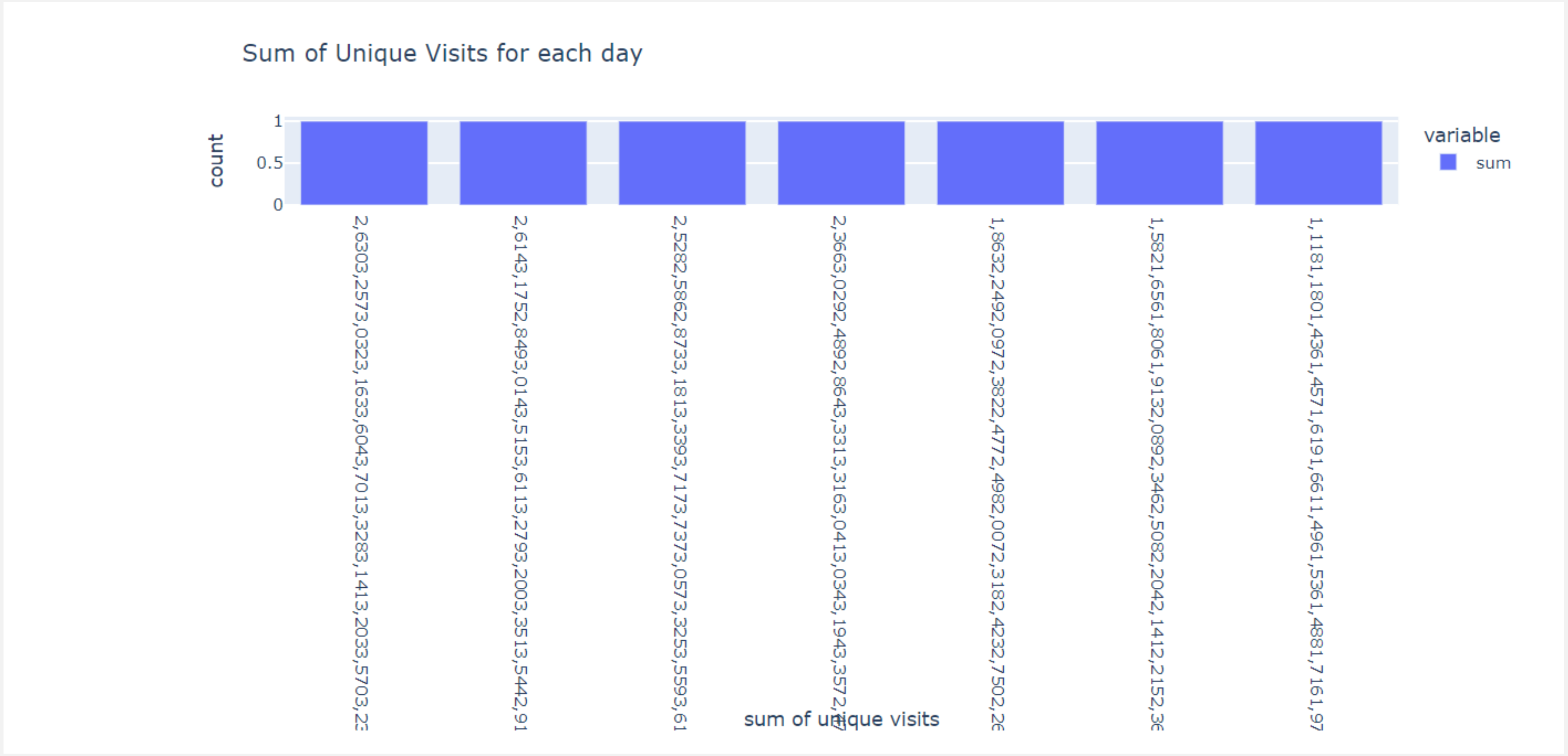
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('webtraffic-analysis.csv')

df = df.dropna()

df = df.drop_duplicates()

df.to_csv('webtraffic-analysis.csv',index=False)
```

```
import plotly.express as px

px.histogram(df,x='Unique.Visits',color='Day',title='unique Visits for Each Day')
```

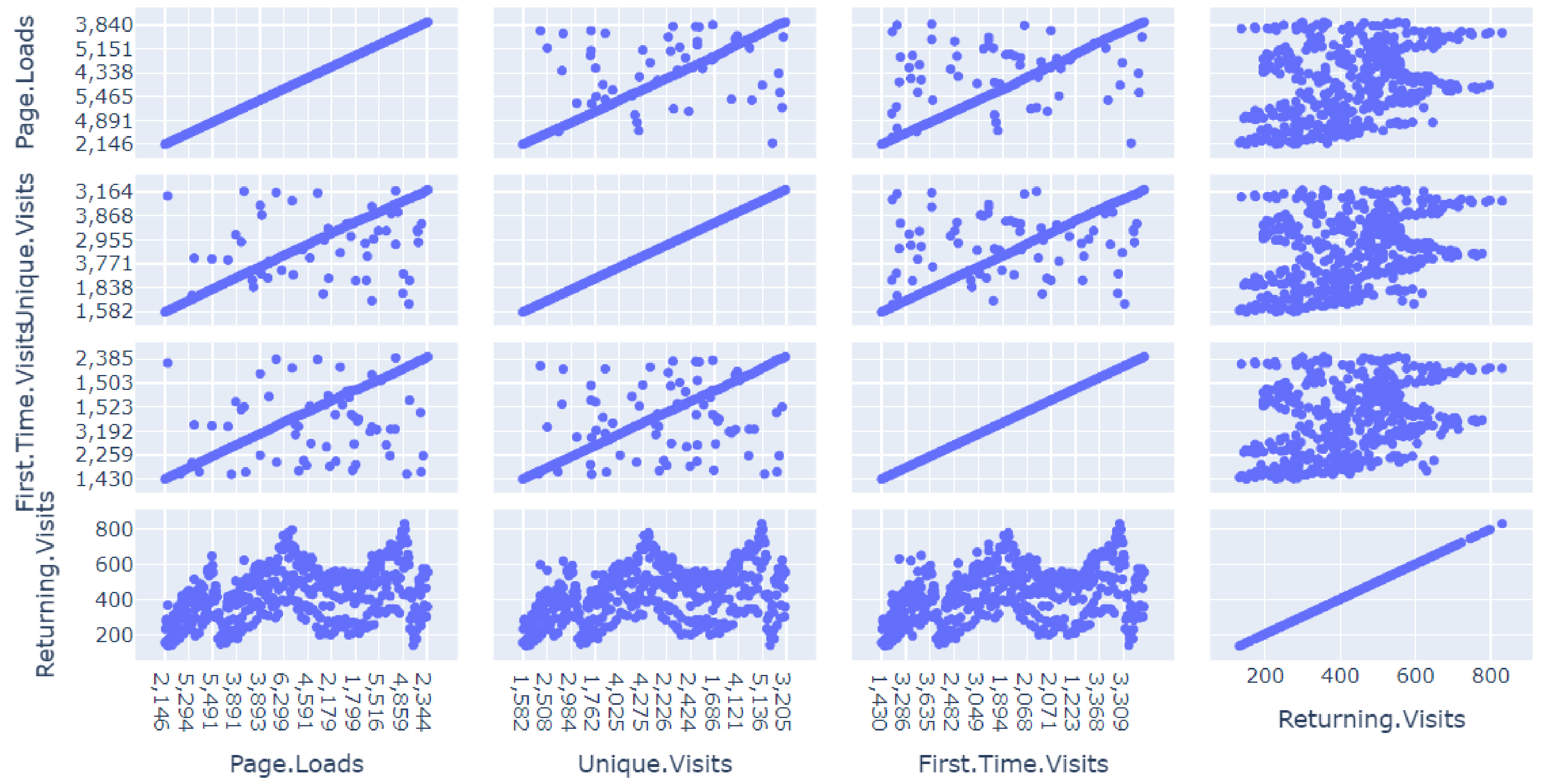


```
day_imp=df.groupby(['Day'])['Unique.Visits'].agg(['sum']).sort_values(by='sum',ascending=False)
px.bar(day_imp,labels={'value':'sum of unique visits'},title='Sum of Unique Visits for each day')

px.histogram(df,x='Date',y='Unique.Visits',color='Day',title='Sum of Unique visits for each day over Time')

px.density_heatmap(df,x='Date',y=['Page.Loads','Unique.visits','First.Time.Visit','Returning.Visits']
                ,marginal_x="histogram",marginal_y="histogram")

px.scatter_matrix(df[['Page.Loads','Unique.Visits','First.Time.Visits','Returning.Visits']])

px.line(df,x='Data',y=['page.Loads','Unique.Visits','First.Time.Visits','Returning.Visits'],
      labels={'value':'Visits'}
      ,title='page Loads & visitors over Time')
```

```
day_imp=df.groupby(['Day'])['Unique.Visits'].agg(['sum']).sort_values(by='sum',ascending=False)
px.bar(day_imp,labels={'value':'sum of unique visits'},title='Sum of Unique Visits for each day')
```

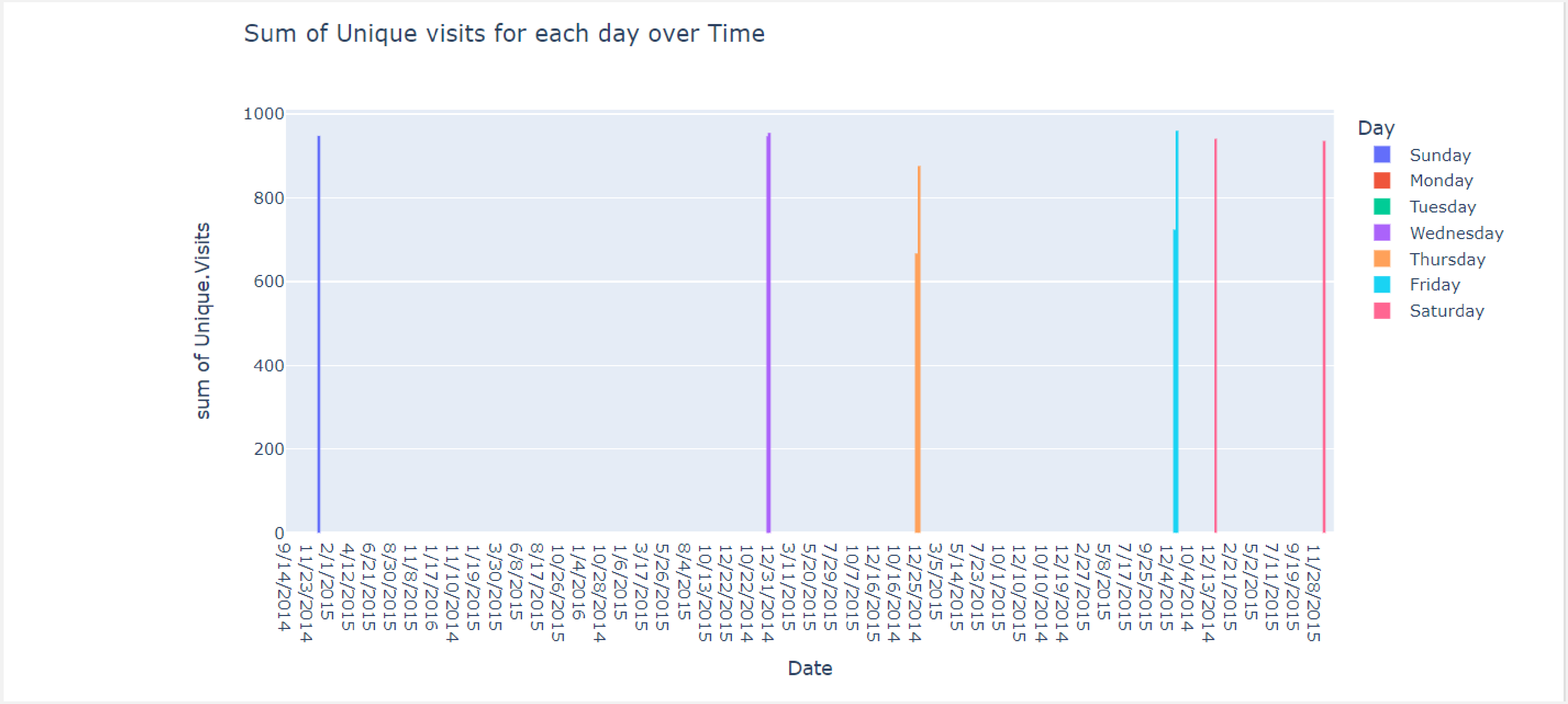


```
px.scatter_matrix(df[['Page.Loads','Unique.Visits','First.Time.Visits','Returning.Visits']])
```

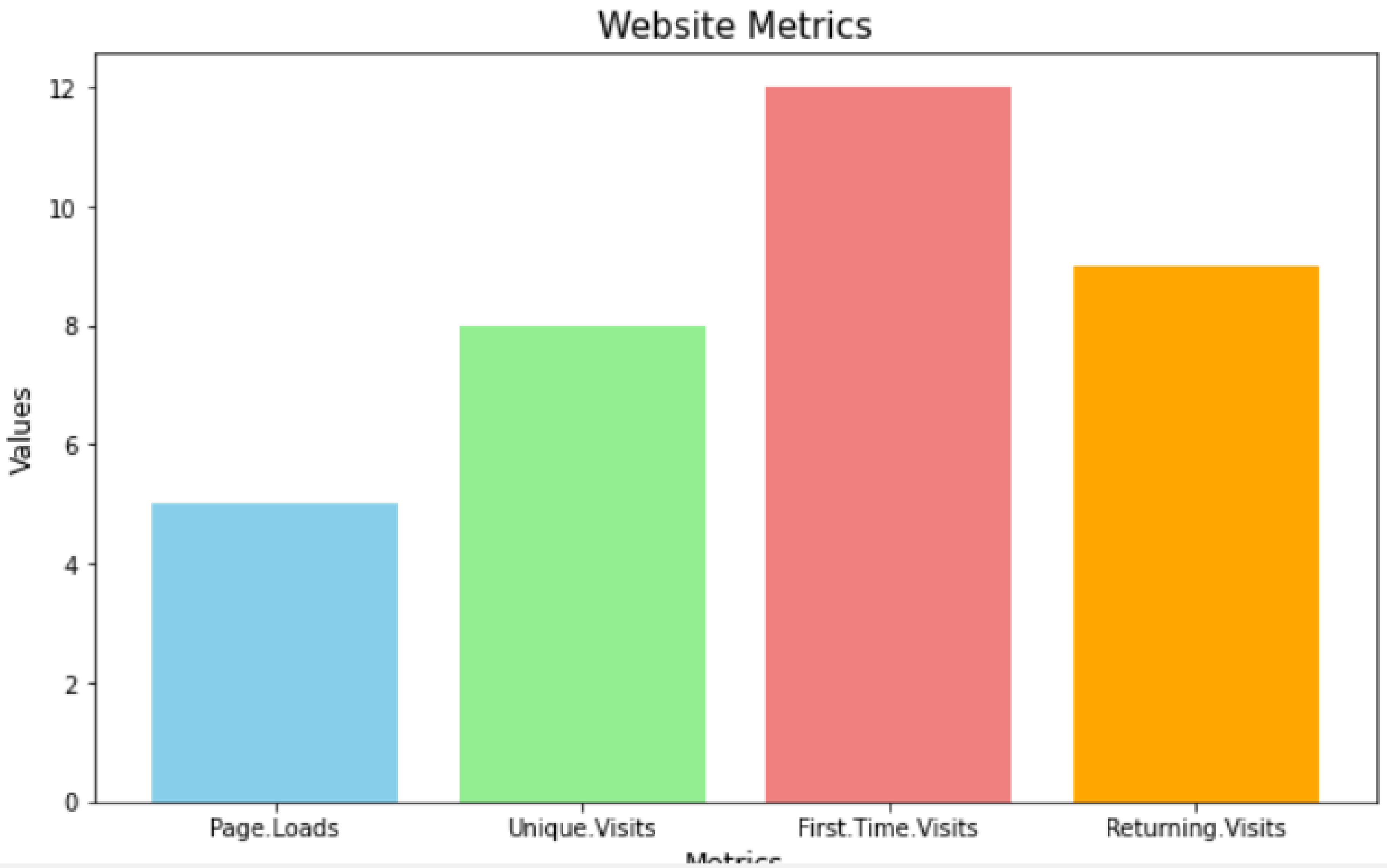


```
import matplotlib.pyplot as plt

# Data
categories = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']
values = [5, 10, 15, 20]

# Creating a scatter plot using Matplotlib
plt.figure(figsize=(8, 6))
plt.scatter(categories, values, color='b', marker='o')
plt.title('Website Metrics Scatter Plot', fontsize=15)
plt.xlabel('Metrics', fontsize=12)
plt.ylabel('Values', fontsize=12)
plt.show()
```

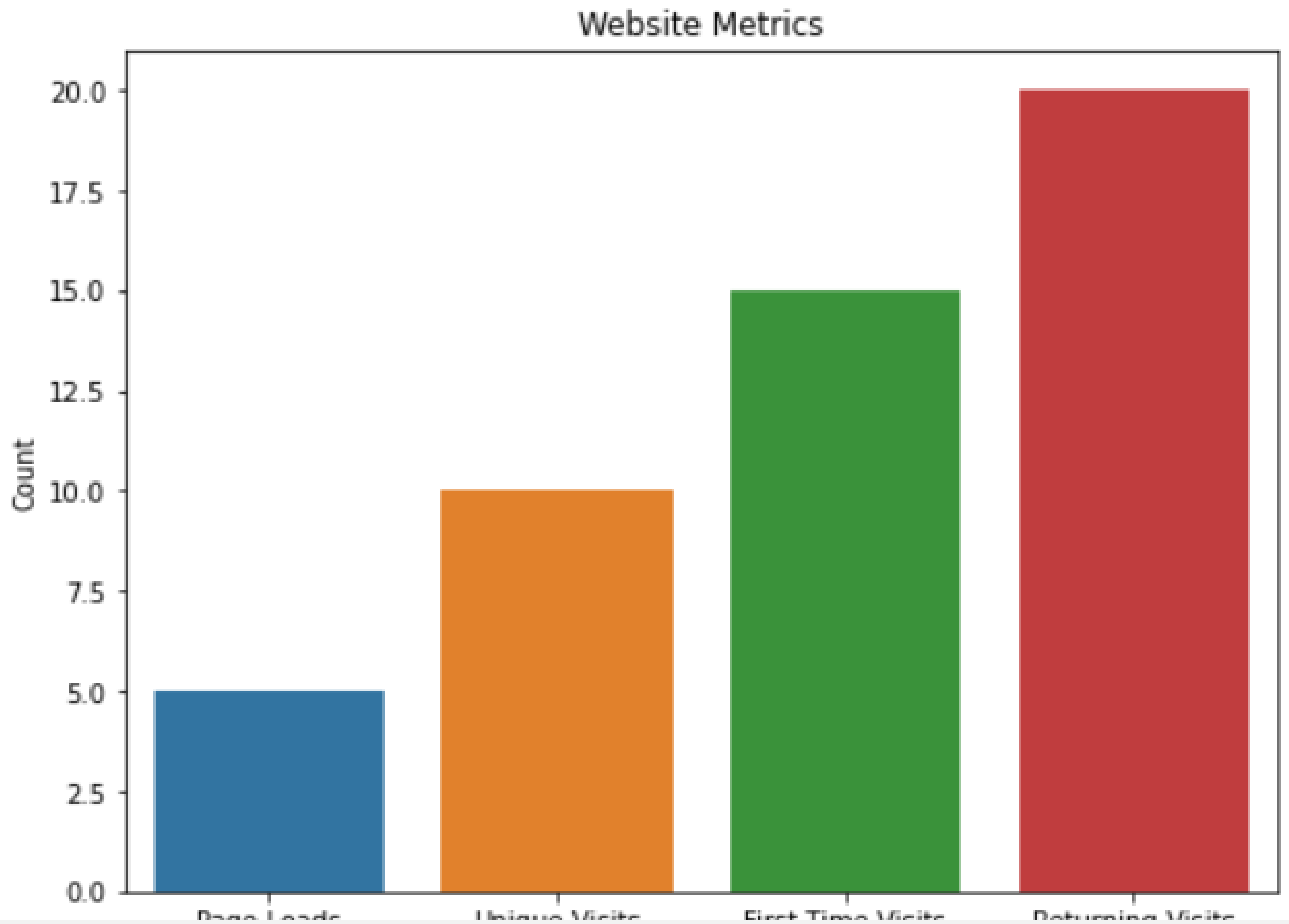
```
px.histogram(df,x='Date',y='Unique.Visits',color='Day',title='Sum of Unique visits for each day over Time')
```



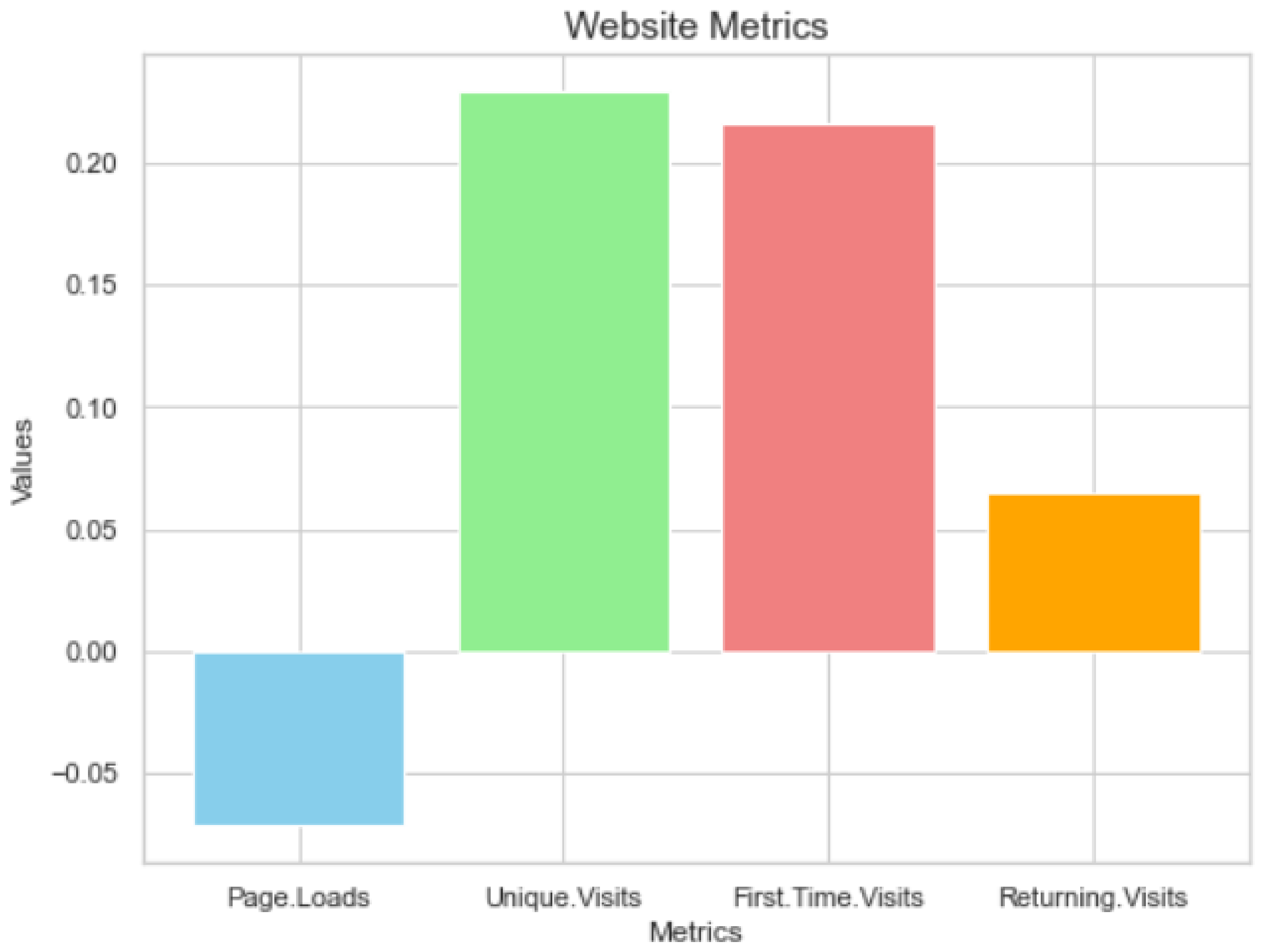
```
import plotly.express as px
px.histogram(df,x='Unique.Visits',color='Day',title='unique Visits for Each Day')


px.histogram(df,x='Date',y='Unique.Visits',color='Day',title='Sum of Unique visits for each day over Time')


import matplotlib.pyplot as plt

# Data
categories = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']
values = [5, 8, 12, 9]

# Creating a bar plot
plt.figure(figsize=(10,6))
plt.bar(categories, values, color=['skyblue', 'lightgreen', 'lightcoral', 'orange'])
plt.xlabel('Metrics', fontsize=12)
plt.ylabel('Values', fontsize=12)
plt.title('Website Metrics', fontsize=15)
plt.show()
```


Website Metrics
Values
0
2
4
6
8
10
12
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits
Metrics

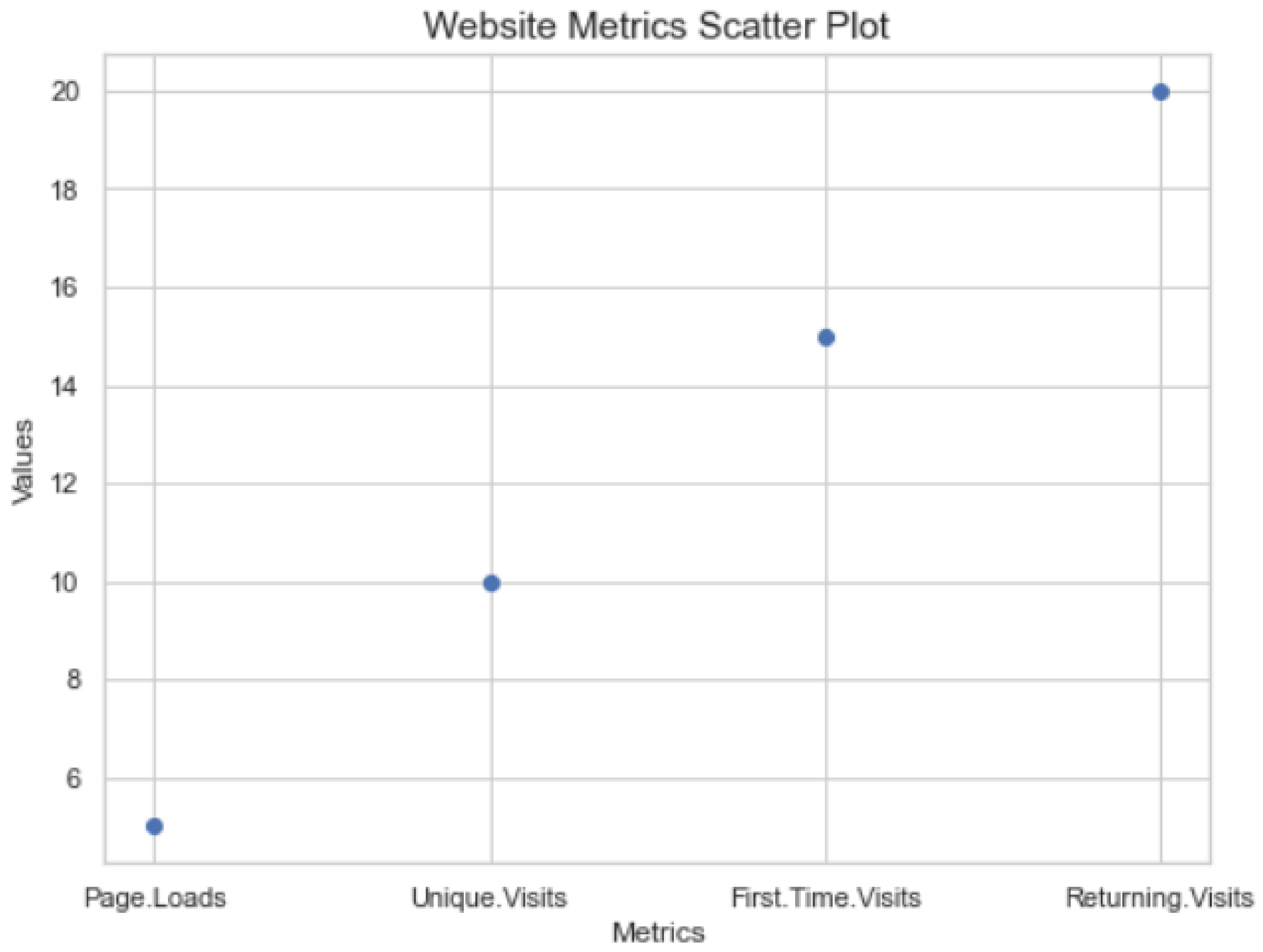
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
categories = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']
values = [5, 10, 15, 20]

# Create a bar plot using Seaborn
plt.figure(figsize=(8, 6))
sns.barplot(x=categories, y=values)
plt.title('Website Metrics')
plt.xlabel('Metrics')
plt.ylabel('Count')
plt.show()
```


Website Metrics
Count
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Page Loads
Unique Visits
First Time Visits
Returning Visits

```
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification

# Generating sample data using sklearn
data, _ = make_classification(n_samples=100, n_features=4, random_state=0)

# Assuming the data is represented by the variables 'Page.Loads', 'Unique.Visits', 'First.Time.Visits', and
'Returning.Visits'
categories = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']
values = data[0]  # Using the generated data for illustration purposes

# Creating a simple bar plot using Matplotlib
plt.figure(figsize=(8, 6))
plt.bar(categories, values, color=['skyblue', 'lightgreen', 'lightcoral', 'orange'])
plt.title('Website Metrics', fontsize=15)
plt.xlabel('Metrics', fontsize=12)
plt.ylabel('Values', fontsize=12)
plt.show()
```


Website Metrics
Values
Metrics
0.20
0.15
0.10
0.05
0.00
−0.05
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits

```
import matplotlib.pyplot as plt

# Data
categories = ['Page.Loads', 'Unique.Visits', 'First.Time.Visits', 'Returning.Visits']
values = [5, 10, 15, 20]

# Creating a scatter plot using Matplotlib
plt.figure(figsize=(8, 6))
plt.scatter(categories, values, color='b', marker='o')
plt.title('Website Metrics Scatter Plot', fontsize=15)
plt.xlabel('Metrics', fontsize=12)
plt.ylabel('Values', fontsize=12)
plt.show()
```


Website Metrics Scatter Plot
Values
20
18
16
14
12
10
8
6
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits
Metrics

THIS PROJECT IS PRESENTED BY:

Soalwin Thomas

Sri jaya shankaran.R

Thamizh Selvan.S

Soleswaran.N

Vinothan.V

THANK YOU!