Mini Project

Batch: C1-1

Project Title: Consumer Behaviour Analysis

Group Members Names and SAP ID:

Cleon Lopes: 60004220071 Bhumil Shah: 60004220063

Purpose of Project:

Customer Behavior Analysis is a process that involves examining and understanding how customers interact with a business, product, or service. This analysis helps organizations make informed decisions, tailor their strategies, and enhance customer experiences.

Customer Behaviour Analysis: (Process We Followed)

Customer Behavior Analysis is a valuable process that empowers businesses to make data-driven decisions, enhance customer experiences, and remain competitive in a dynamic market. Below is the process we can follow for the task of Customer Behaviour Analysis:

1. Collect data related to customer interactions. It can include purchase history, website visits, social media engagement, customer feedback, and more.
2. Identify and address data inconsistencies, missing values, and outliers to ensure the data's quality and accuracy
3. Calculate basic statistics like mean, median, and standard deviation to summarize data.
4. Create visualizations such as histograms, scatter plots, and bar charts to explore trends, patterns, and anomalies in the data.
5. Use techniques like clustering to group customers based on common behaviours or characteristics.

So, the process starts with collecting data based on customer behaviour on a platform. Our team found an ideal dataset for this purpose.

Mini Project

Code:

```
import pandas as pd

import matplotlib.pyplot as plt
import numpy as np

data = pd.read_csv("ecommerce_customer_data.csv")
print(data.head())
print("\n")
```

```
PROBLEMS 8  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Bhumil Shah\OneDrive\Desktop\Python1> python -u "c:\Users\Bhumil Shah\OneDrive\Desktop\Python1\project.py"
   User_ID  Gender  Age   Location Device_Type  Product_Browsing_Time  Total_Pages_Viewed  Items_Added_to_Cart  Total_Purchases
0        1  Female   23  Ahmedabad      Mobile                     60                  30                    1                0
1        2    Male   25    Kolkata      Tablet                     30                  38                    9                4
2        3    Male   32  Bangalore     Desktop                     37                  13                    5                0
3        4    Male   35      Delhi      Mobile                      7                  20                   10                3
4        5    Male   27  Bangalore      Tablet                     35                  20                    8                2
```

```
# column info in the dataset
print(data.info())
print("\n")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   User_ID                 500 non-null    int64
 1   Gender                  500 non-null    object
 2   Age                     500 non-null    int64
 3   Location                500 non-null    object
 4   Device_Type             500 non-null    object
 5   Product_Browsing_Time   500 non-null    int64
 6   Total_Pages_Viewed      500 non-null    int64
 7   Items_Added_to_Cart     500 non-null    int64
 8   Total_Purchases         500 non-null    int64
dtypes: int64(6), object(3)
memory usage: 35.3+ KB
None
```

Mini Project

```
# printing all the columns in dataset
print(data.columns)
print("\n")
```

```
Index(['User_ID', 'Gender', 'Age', 'Location', 'Device_Type',
       'Product_Browsing_Time', 'Total_Pages_Viewed', 'Items_Added_to_Cart',
       'Total_Purchases'],
      dtype='object')
```

```
# Summary statistics for numeric columns
numeric_summary = data.describe()
print(numeric_summary)
print("\n")
```

	User_ID	Age	Product_Browsing_Time	Total_Pages_Viewed	Items_Added_to_Cart	Total_Purchases
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000
mean	250.500000	26.276000	30.740000	27.182000	5.150000	2.464000
std	144.481833	5.114699	15.934246	13.071596	3.203127	1.740909
min	1.000000	18.000000	5.000000	5.000000	0.000000	0.000000
25%	125.750000	22.000000	16.000000	16.000000	2.000000	1.000000
50%	250.500000	26.000000	31.000000	27.000000	5.000000	2.000000
75%	375.250000	31.000000	44.000000	38.000000	8.000000	4.000000
max	500.000000	35.000000	60.000000	50.000000	10.000000	5.000000

```
# Summary for non-numeric columns
categorical_summary = data.describe(include='object')
print(categorical_summary)
print("\n")
```

	Gender	Location	Device_Type
count	500	500	500
unique	2	8	3
top	Male	Kolkata	Mobile
freq	261	71	178

Mini Project

```
# look if your data contains any missing values or not:
print(data.isnull().sum())
print("\n")
```

```
User_ID                 0
Gender                  0
Age                     0
Location                0
Device_Type             0
Product_Browsing_Time   0
Total_Pages_Viewed      0
Items_Added_to_Cart     0
Total_Purchases         0
dtype: int64
```

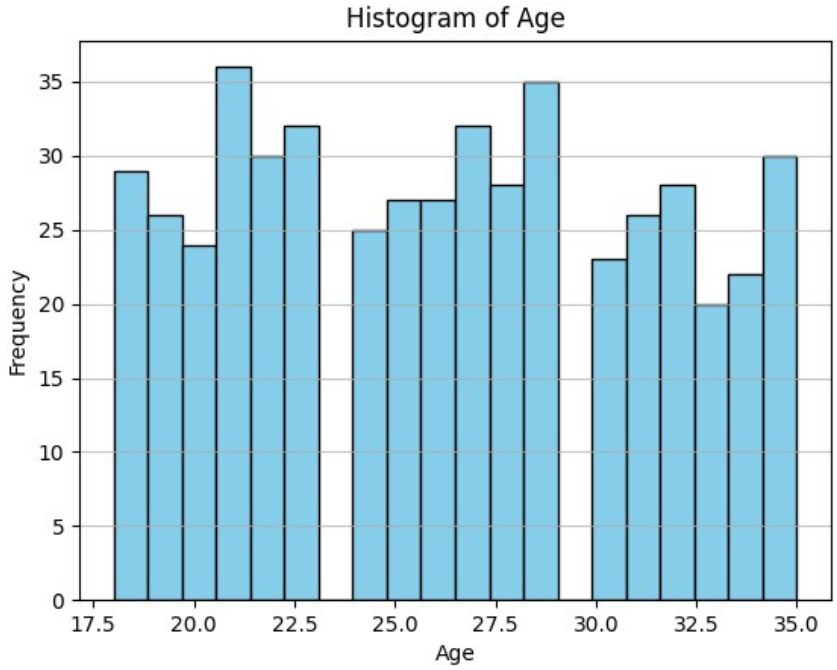
```
# print the user id, the type of item purchased by the user along with the number of items purchased
data1 = ['User_ID','Device_Type','Total_Purchases']
condition = data["Total_Purchases"] > 0
print(data.loc[condition, data1])
```

```
     User_ID Device_Type  Total_Purchases
1          2      Tablet                4
3          4      Mobile                3
4          5      Tablet                2
5          6      Mobile                4
7          8      Mobile                2
..       ...         ...              ...
494      495     Desktop                1
496      497     Desktop                5
497      498     Desktop                3
498      499     Desktop                4
499      500      Mobile                4
```

```
# Calculate churn rate -> the annual percentage rate at which customers stop subscribing to a service / employees leave a job.
data['Churned'] = data['Total_Purchases'] == 0
churn_rate = data['Churned'].mean()
print(f"Churn Rate:{churn_rate}")
```

```
Churn Rate:0.198
```

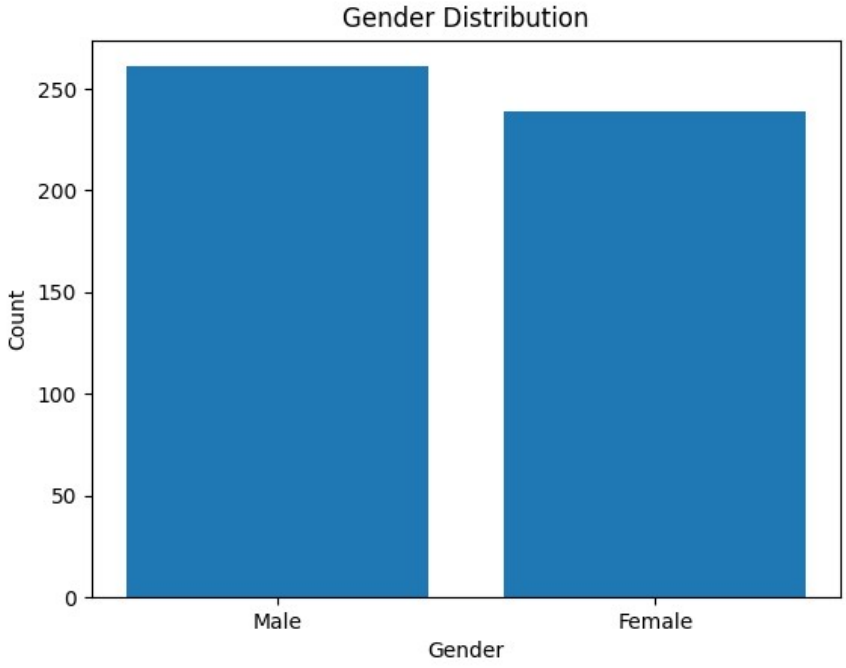
```
# Histogram for 'Age'
plt.hist(data['Age'], bins=20, color='skyblue', edgecolor='black')
plt.title('Histogram of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(axis='y', alpha=0.75)
plt.show()
```



- The histogram will show you the distribution of ages in your dataset. You can quickly identify whether your users predominantly belong to a specific age group or if the distribution is spread across various age ranges.

- By looking at the histogram, you can identify the most common age groups among your users. This information is crucial for targeted marketing or tailoring your products/services to specific age demographics. Investigating unusual spikes or gaps may be worth understanding the underlying reasons

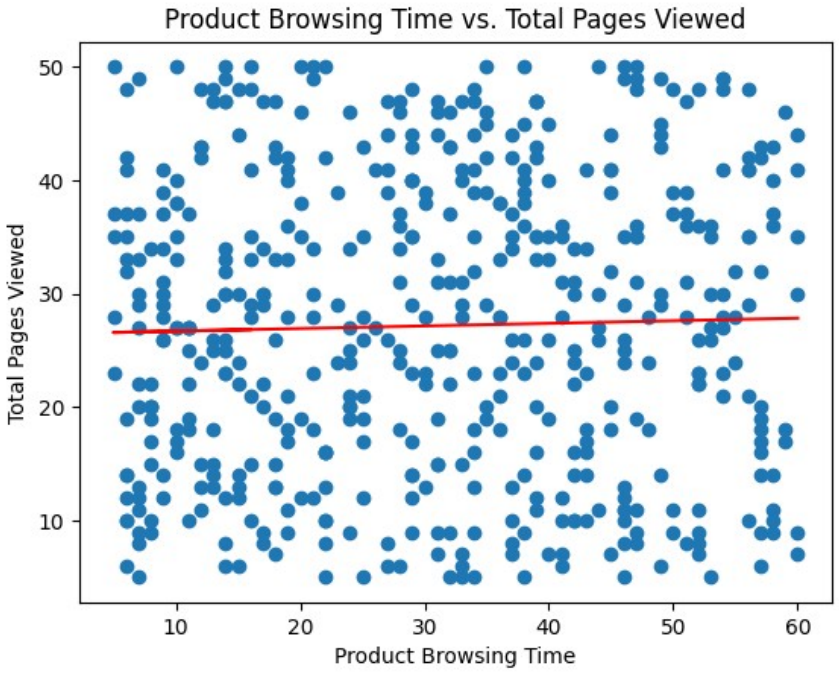
```
# Bar chart for 'Gender'
gender_counts = data['Gender'].value_counts().reset_index()
print(gender_counts)
gender_counts.columns = ['Gender', 'Count'] # rename colums
plt.bar(gender_counts['Gender'], gender_counts['Count']) #data
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```



- The bar chart will give you a clear visual representation of the distribution of genders in your dataset. This basic demographic information is fundamental for understanding the composition of your user base.

- Knowing the gender distribution allows you to tailor your marketing strategies more effectively. For example, if your data shows a significant majority of one gender, you might customize your advertising or promotional content to better resonate with that particular group.

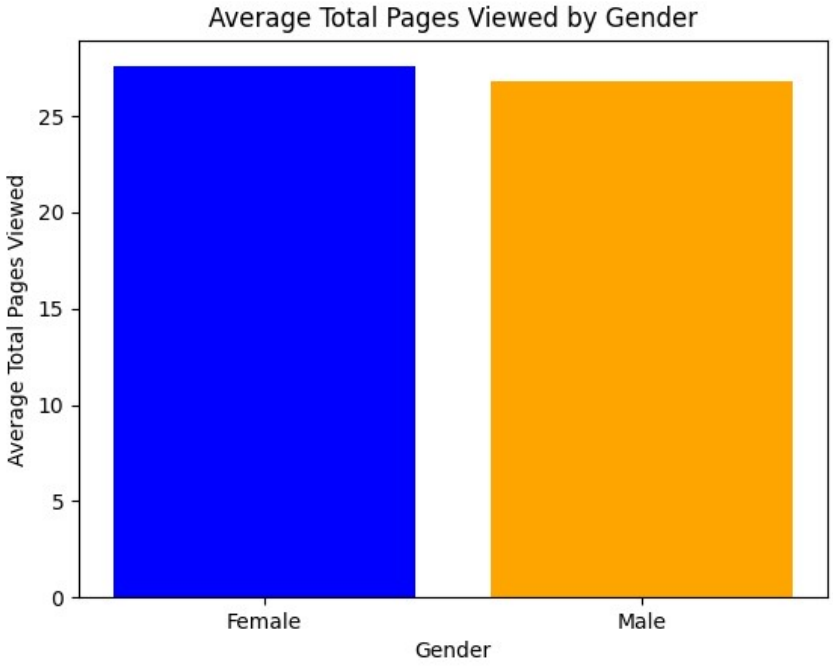
```
# 'Product_Browsing_Time' vs 'Total_Pages_Viewed'
plt.scatter(data['Product_Browsing_Time'], data['Total_Pages_Viewed'])
# Linear regression
m, b = np.polyfit(data['Product_Browsing_Time'], data['Total_Pages_Viewed'], 1)
plt.plot(data['Product_Browsing_Time'], m * data['Product_Browsing_Time'] + b, color='red')
plt.title('Product Browsing Time vs. Total Pages Viewed')
plt.xlabel('Product Browsing Time')
plt.ylabel('Total Pages Viewed')
plt.show()
```



- Outliers in the scatter plot can highlight unusual or unexpected patterns in your data. For example, users with exceptionally long browsing times and low total pages viewed or vice versa might stand out. Investigating these outliers can provide insights into unique user behaviors or potential issues with your platform.

- Analyzing the scatter plot can help you assess the effectiveness of your content. If users with longer browsing times tend to view more pages, it might suggest that certain types of content are keeping users engaged. This information can guide content creation and optimization efforts.

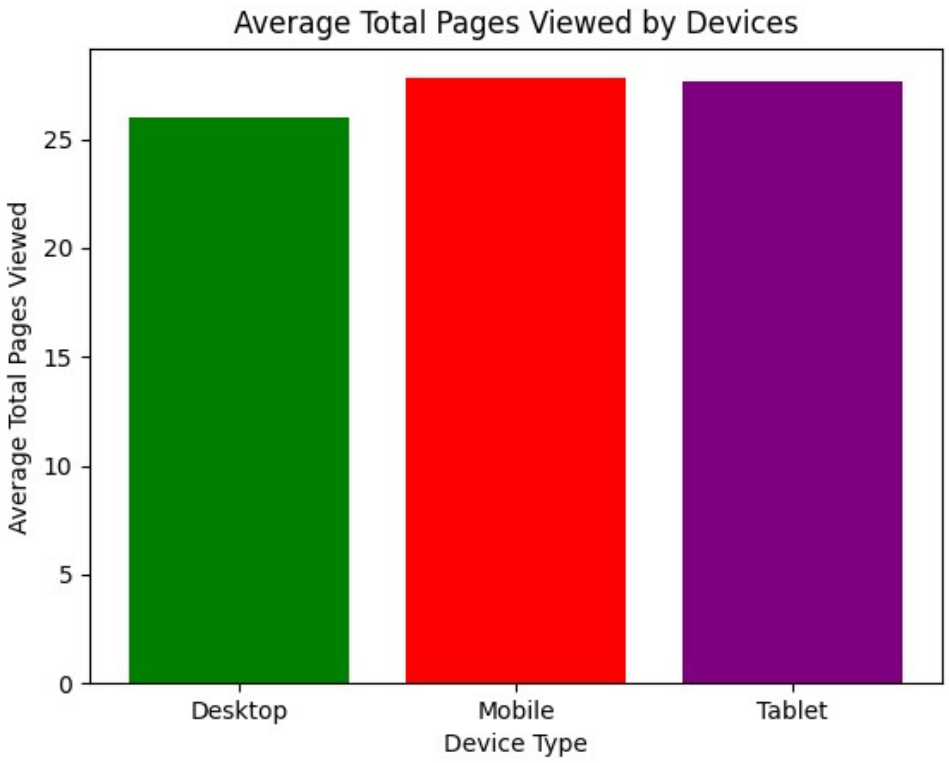
```
# # Grouped Analysis
# Average Total Pages Viewed by Gender
gender_grouped = data.groupby('Gender')['Total_Pages_Viewed'].mean().reset_index()
plt.bar(gender_grouped['Gender'],gender_grouped['Total_Pages_Viewed'],color=['blue', 'orange'])
plt.title('Average Total Pages Viewed by Gender')
plt.xlabel('Gender')
plt.ylabel('Average Total Pages Viewed')
plt.show()
```



- Over time, changes in the average total pages viewed by gender can indicate emerging trends. Monitoring these trends can help you adapt your content and user experience strategies to stay aligned with the evolving preferences of your audience.

- Comparing gender-based average total pages viewed is a valuable performance metric. It can be used to track the success of marketing campaigns, content updates, or user interface changes, allowing you to make data-driven decisions for optimization.

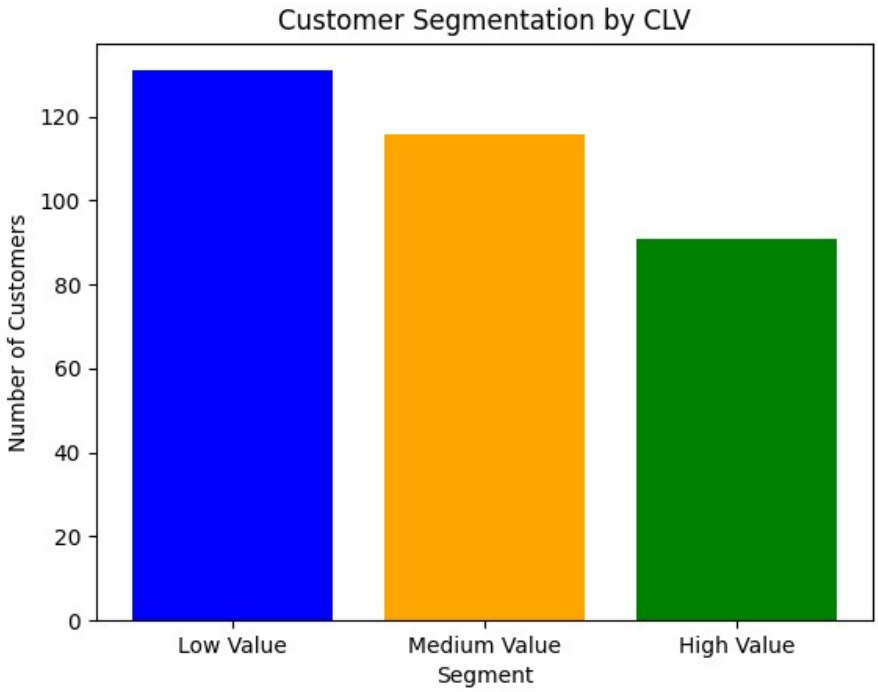
```
# Average Total Pages Viewed by Devices
devices_grouped = data.groupby('Device_Type')['Total_Pages_Viewed'].mean().reset_index()
plt.bar(devices_grouped['Device_Type'], devices_grouped['Total_Pages_Viewed'], color=['green',
'red', 'purple'])
plt.title('Average Total Pages Viewed by Devices')
plt.xlabel('Device Type')
plt.ylabel('Average Total Pages Viewed')
plt.show()
```



- You can identify if there are significant differences in user engagement based on the type of device (e.g., desktop, mobile, tablet). For example, if users on one type of device tend to view more pages on average, it could indicate a preference for a specific platform.

- Insights from this analysis can guide improvements in the user experience for specific devices. For instance, if mobile users have lower average pages viewed, it could prompt enhancements to the mobile interface or user journey.

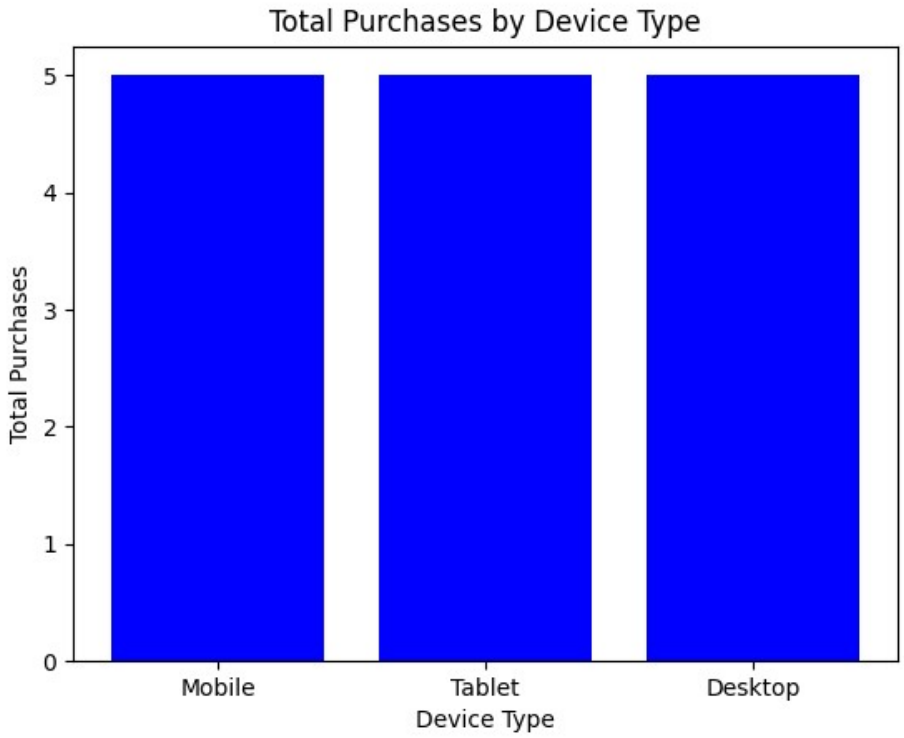
```
# CLV -> Customer Lifetime Value -> Customer Lifetime Value = (Customer Value * Average Customer Lifespan)
data['CLV'] = (data['Total_Purchases'] * data['Total_Pages_Viewed']) / data['Age']
data['Segment'] = pd.cut(data['CLV'], bins=[1, 2.5, 5, float('inf')],
                         labels=['Low Value', 'Medium Value', 'High Value'])
segment_counts = data['Segment'].value_counts().reset_index()
segment_counts.columns = ['Segment', 'Count']
plt.bar(segment_counts['Segment'], segment_counts['Count'], color=['blue', 'orange', 'green'])
plt.title('Customer Segmentation by CLV')
plt.xlabel('Segment')
plt.ylabel('Number of Customers')
plt.show()
```



- Understanding the CLV of different customer segments can guide resource allocation. You may choose to invest more in acquiring or retaining high-value customers, as they contribute more to the overall revenue over time.

- CLV is closely tied to customer retention. By analyzing the CLV bar chart, you can identify segments with lower CLV and work on strategies to improve customer retention in those segments.

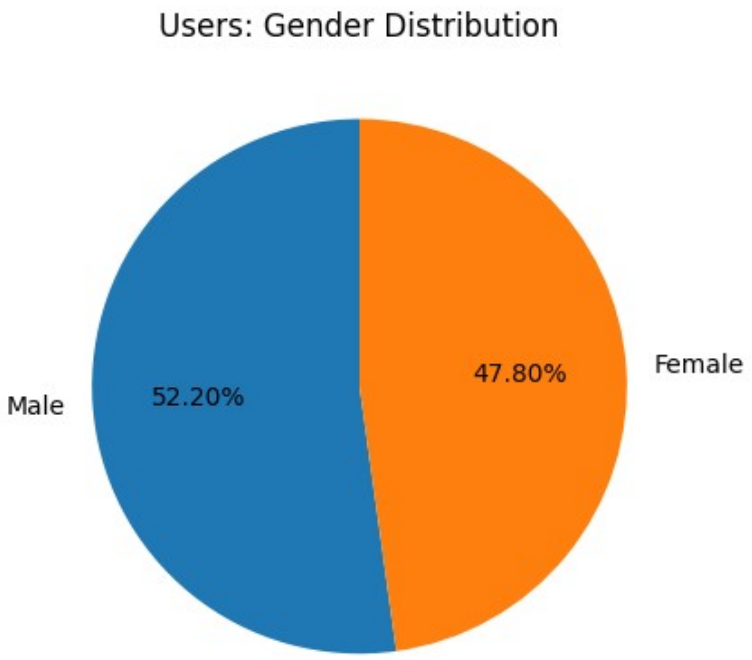
```
# DeviceType vs total purchase graph
plt.bar(data['Device_Type'], data['Total_Purchases'], color=['blue'])
plt.title('Total Purchases by Device Type')
plt.xlabel('Device Type')
plt.ylabel('Total Purchases')
plt.show()
```



- By comparing the total purchases across different devices, you can calculate and analyze the conversion rates for each device type. This can help you identify areas for improvement in the conversion funnel for specific devices.

- The bar graph can visually represent which types of devices (e.g., desktop, mobile, tablet) are more commonly used by your customers for making purchases. This insight can help you optimize your platform for the most preferred devices.

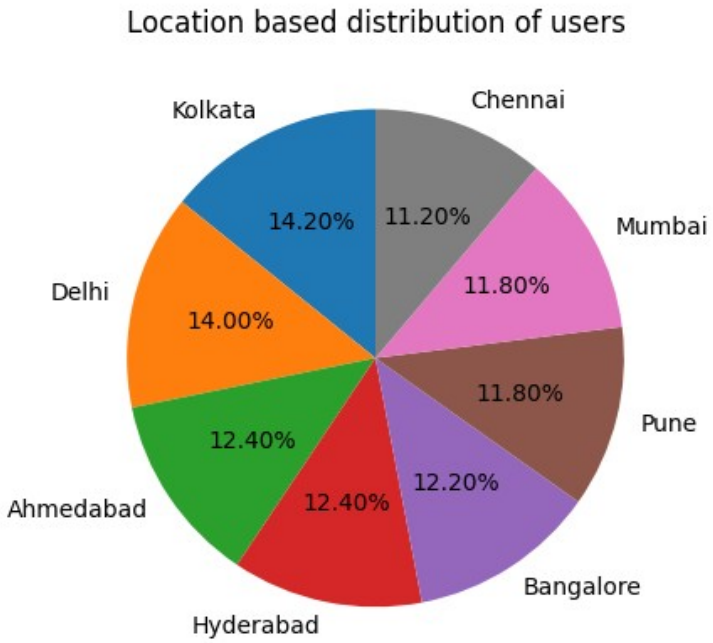
```
# Gender Distribution Pie Chart
gender_distribution = data['Gender'].value_counts()
plt.pie(gender_distribution, labels=gender_distribution.index,autopct='%1.2f%%', startangle=90)
plt.title("Users: Gender Distribution")
plt.show()
```



- The pie chart provides a quick and visually intuitive representation of the gender distribution. This demographic information is fundamental for understanding the composition of your audience or user base.

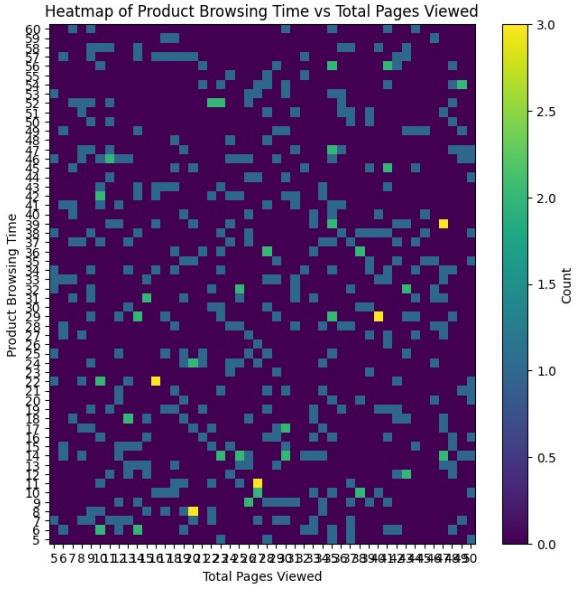
- A gender distribution pie chart can be used to assess diversity and inclusion within an organization or community. Monitoring and improving gender balance is an essential aspect of promoting diversity.

```
# Location Distribution Pie Chart
location_distribution = data['Location'].value_counts()
plt.pie(location_distribution,labels=location_distribution.index,autopct="%1.2f%%", startangle=90)
plt.title("Location based distribution of users")
plt.show()
```



- The pie chart can provide a quick overview of the distribution of data across different locations. This is particularly valuable when dealing with user bases, customers, or other entities that are spread across geographic regions.
- If your analysis involves regional considerations, a location pie chart can help identify which regions contribute the most to your dataset. This information is useful for businesses or organizations looking to target specific areas or adapt strategies based on regional differences.

```
# heat map
heatmap_data = data.pivot_table(index='Product_Browsing_Time', columns='Total_Pages_Viewed',
values='User_ID', aggfunc='count', fill_value=0)
plt.figure(figsize=(10, 8))
plt.imshow(heatmap_data, cmap='viridis', origin='lower', interpolation='none')
plt.colorbar(label='Count')
plt.xlabel('Total Pages Viewed')
plt.ylabel('Product Browsing Time')
plt.xticks(np.arange(len(heatmap_data.columns)), heatmap_data.columns)
plt.yticks(np.arange(len(heatmap_data.index)), heatmap_data.index)
plt.title('Heatmap of Product Browsing Time vs Total Pages Viewed')
plt.show()
```



- By examining the heatmap, you can visually assess whether there's a correlation between product browsing time and total pages viewed. For example, you might observe that longer browsing times tend to be associated with higher total pages viewed, or vice versa.

- If your dataset includes additional categorical variables, you can use a heatmap to segment the data. For instance, you might color the heatmap differently for different user segments, helping you identify variations in the relationship between product browsing time and total pages viewed across these segments.