INTRODUCTION

This dataset describes web traffic on an anonymous webpage. It describes the origin of the traffic, the number of pages viewed, and more. It also contains the conversion rate, the target variable for converting traffic into desired actions (like purchases).

The features are as follows:

- Page Views: The number of pages the user viewed during the session
- Session Duration: The length of the session in minutes
- Bounce Rate: The percentage of visitors who left after visiting a single page. Exact calculation unknown.
- Traffic Source: The way the traffic originated, be it through organic search or through paid ads.
- Time on Page: The amount of time, in seconds, user spent on the specific page as the data was captured.
- Previous Visits: The number of times the current user has visited the page in the past.
- Conversion Rate: The percentage of users during the session that completed a desired interaction. Exact calculation unknown. Target variable.

The dataset was found on Kaggle here:
https://www.kaggle.com/datasets/anthonytherrien/website-traffic

The dataset contains 2000 rows and 7 features.

LOADING DATA

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

```
df = pd.read_csv('/content/website_wata.csv')
```

```
df.head()
```

	Page Views	Session Duration	Bounce Rate	Traffic Source	Time on Page	Previous Visits	Conversion Rate
0	5	11.051381	0.230652	Organic	3.890460	3	1.0
1	4	3.429316	0.391001	Social	8.478174	0	1.0
2	4	1.621052	0.397986	Organic	9.636170	2	1.0
3	5	3.629279	0.180458	Organic	2.071925	3	1.0
4	5	4.235843	0.291541	Paid	1.960654	5	1.0

Exploratory Data Analysis

```
df.shape
```

```
(2000, 7)
```

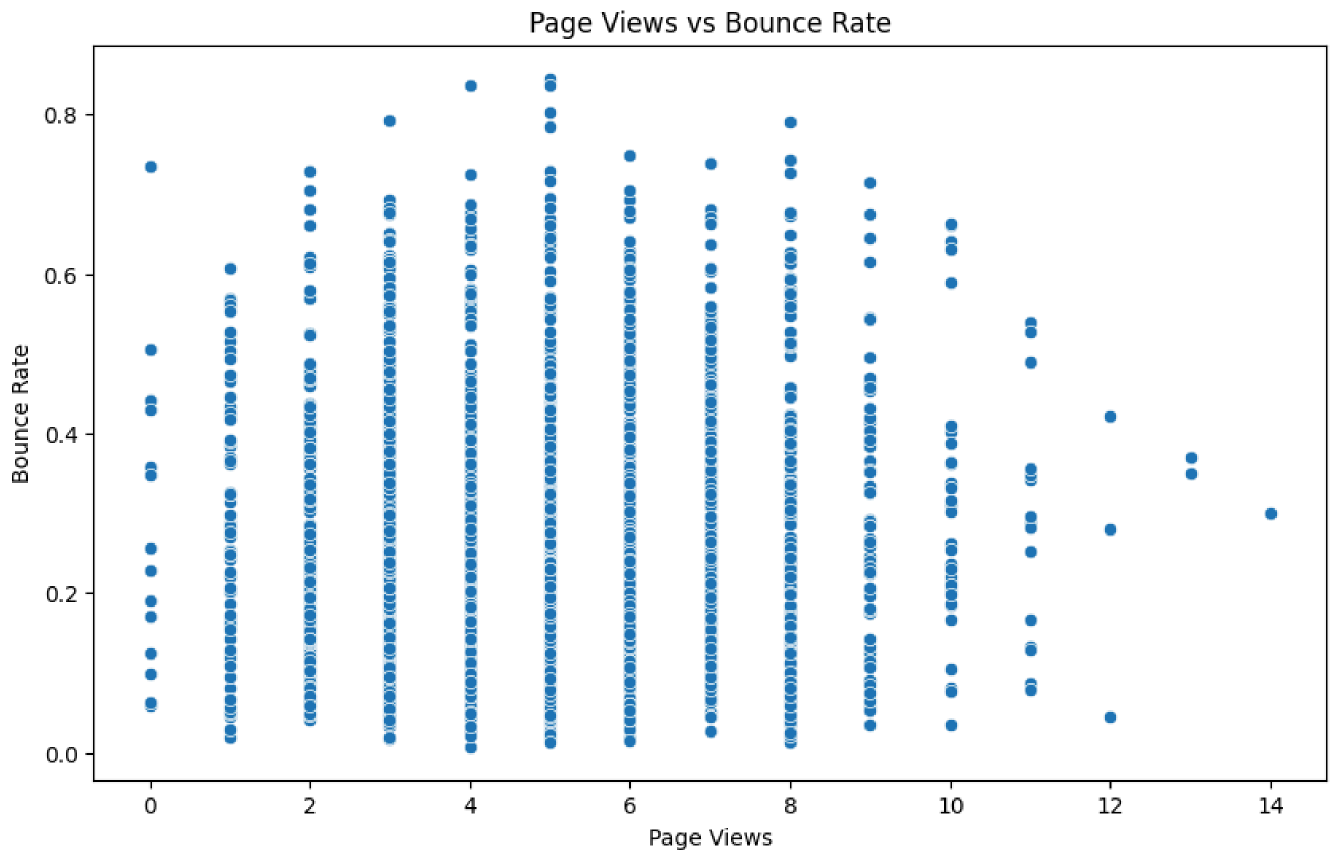
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 7 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Page Views        2000 non-null   int64
 1   Session Duration  2000 non-null   float64
 2   Bounce Rate       2000 non-null   float64
 3   Traffic Source    2000 non-null   object
 4   Time on Page      2000 non-null   float64
 5   Previous Visits   2000 non-null   int64
 6   Conversion Rate   2000 non-null   float64
dtypes: float64(4), int64(2), object(1)
memory usage: 109.5+ KB
```

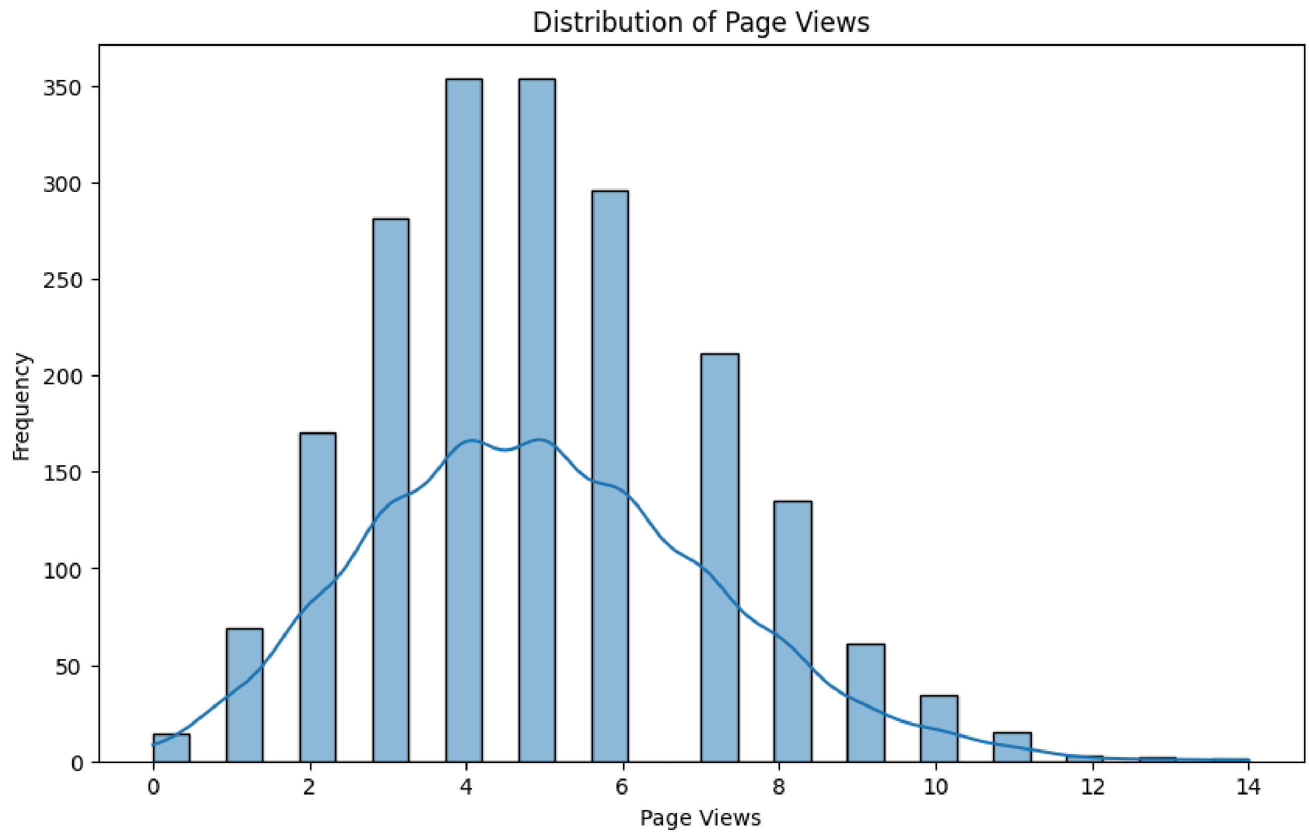
```
df.describe()
```

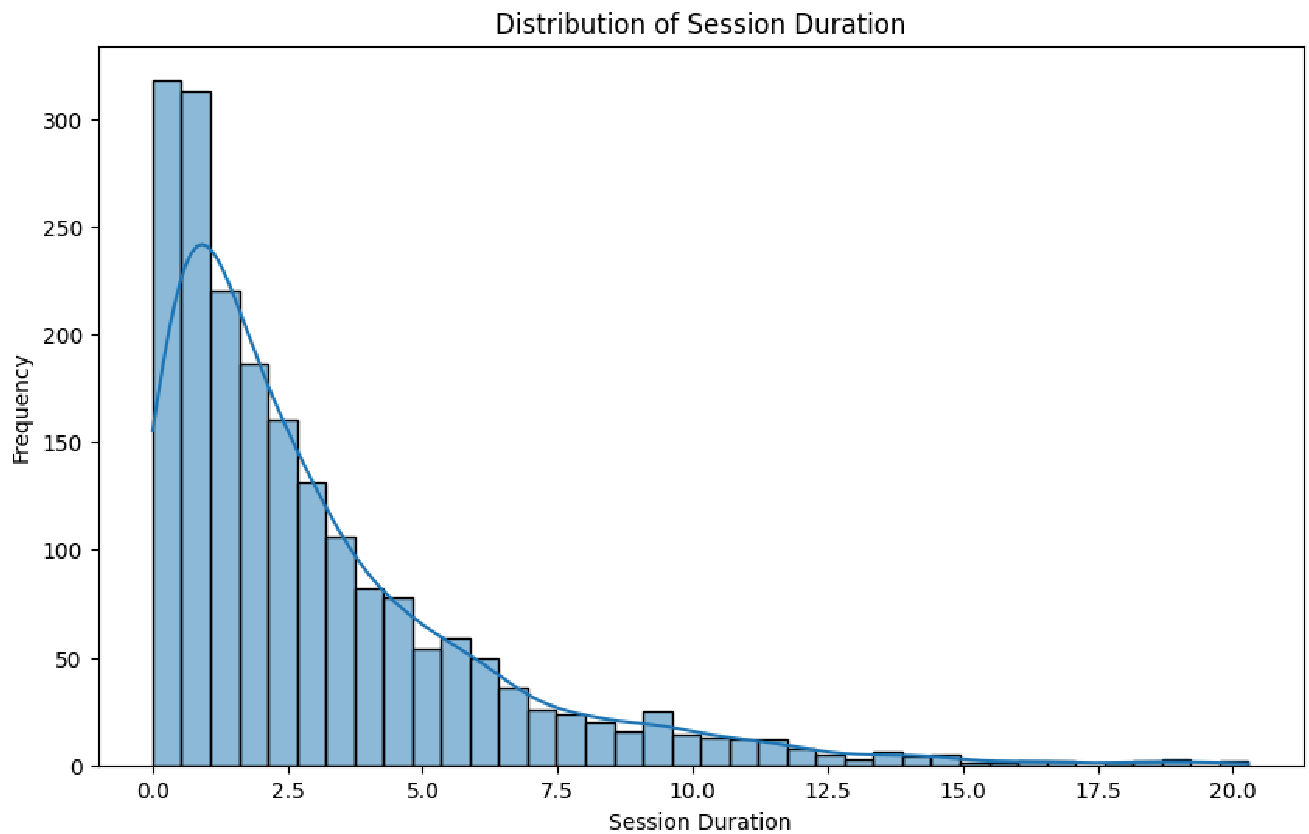
	Page Views	Session Duration	Bounce Rate	Time on Page	Previous Visits	Conversion Rate
count	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	4.950500	3.022045	0.284767	4.027439	1.978500	0.982065
std	2.183903	3.104518	0.159781	2.887422	1.432852	0.065680
min	0.000000	0.003613	0.007868	0.068515	0.000000	0.343665
25%	3.000000	0.815828	0.161986	1.935037	1.000000	1.000000
50%	5.000000	1.993983	0.266375	3.315316	2.000000	1.000000
75%	6.000000	4.197569	0.388551	5.414627	3.000000	1.000000
max	14.000000	20.290516	0.844939	24.796182	9.000000	1.000000

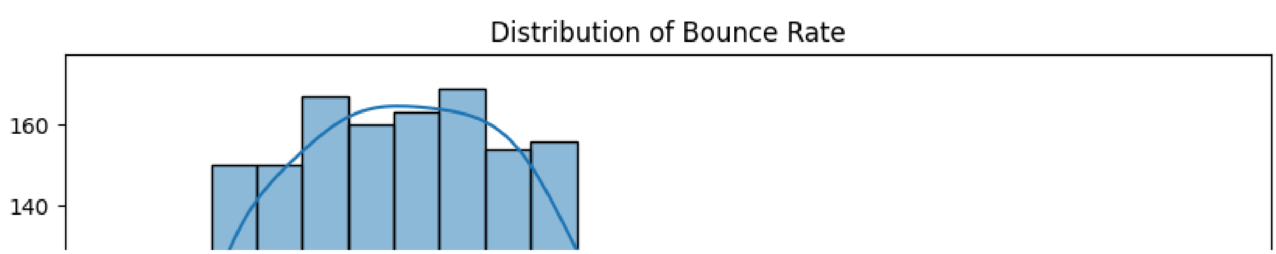
```
# Trying to interpret 'Bounce Rate' by comparing it with Page Views
x = df[['Page Views']]
y = df[['Bounce Rate']]

plt.figure(figsize=(10,6))
sns.scatterplot(x='Page Views', y='Bounce Rate', data=df)
plt.title('Page Views vs Bounce Rate')
plt.xlabel('Page Views')
plt.ylabel('Bounce Rate')
plt.show()
```

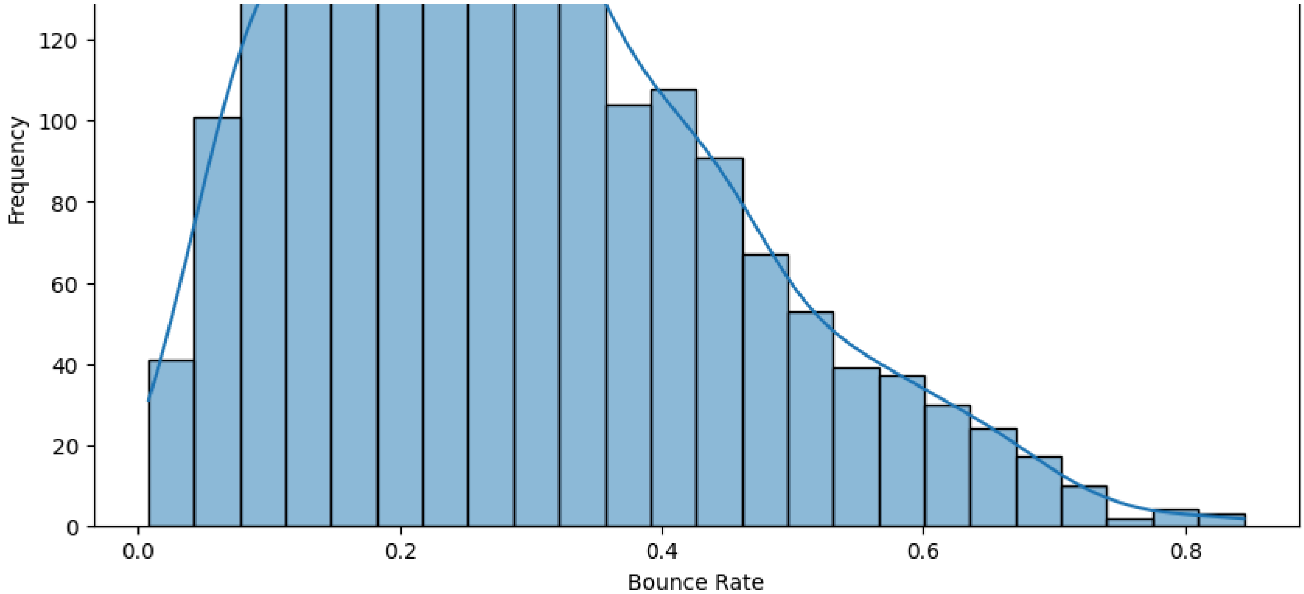


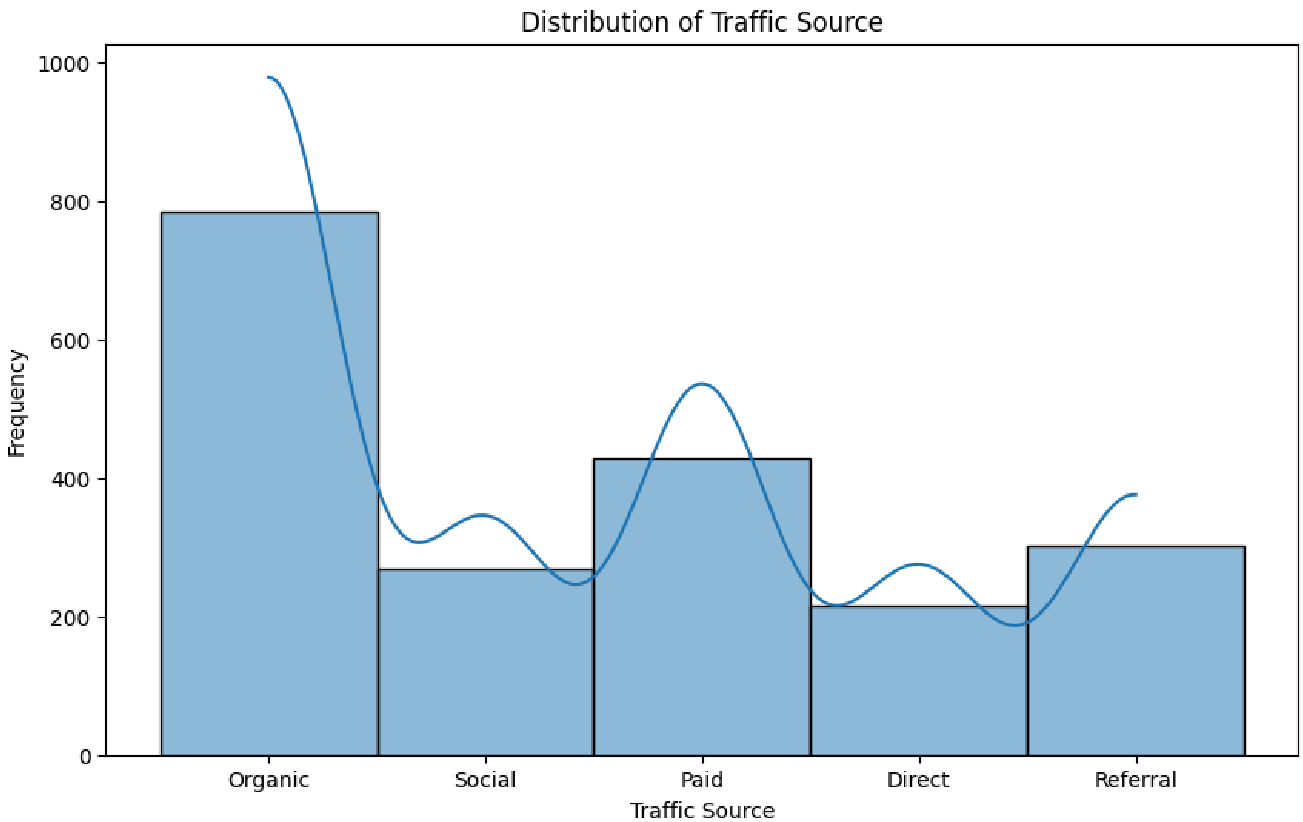

```
for col in df.columns:
  plt.figure(figsize=(10,6))
  sns.histplot(df[col], kde=True)
  plt.title(f'Distribution of {col}')
  plt.xlabel(col)
  plt.ylabel('Frequency')
  plt.show()
```

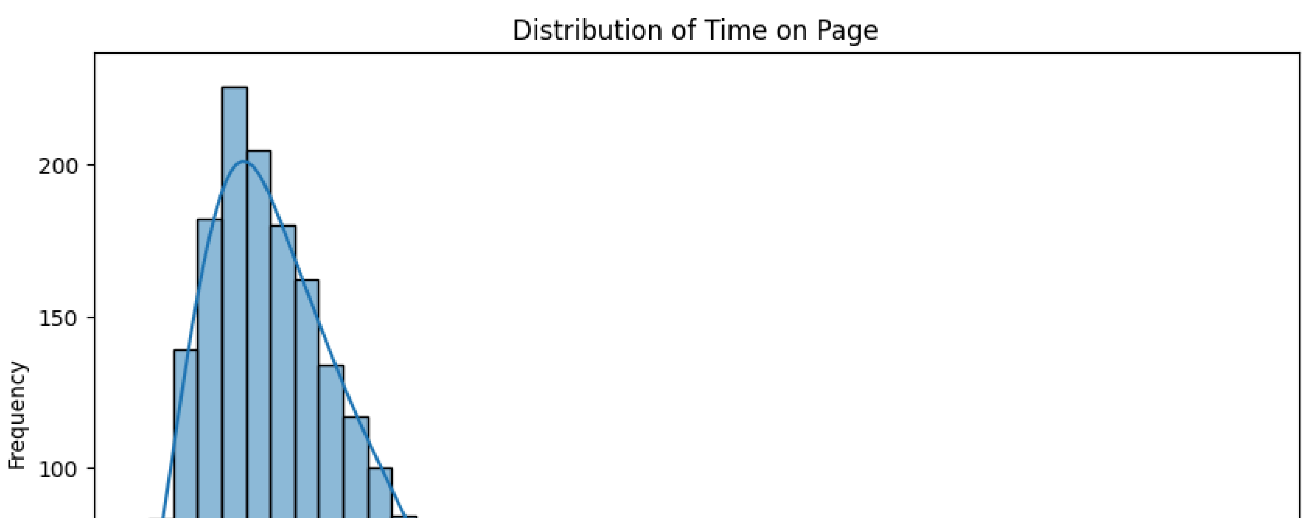

Distribution of Page Views
Frequency
Page Views
0
50
100
150
200
250
300
350
0
2
4
6
8
10
12
14

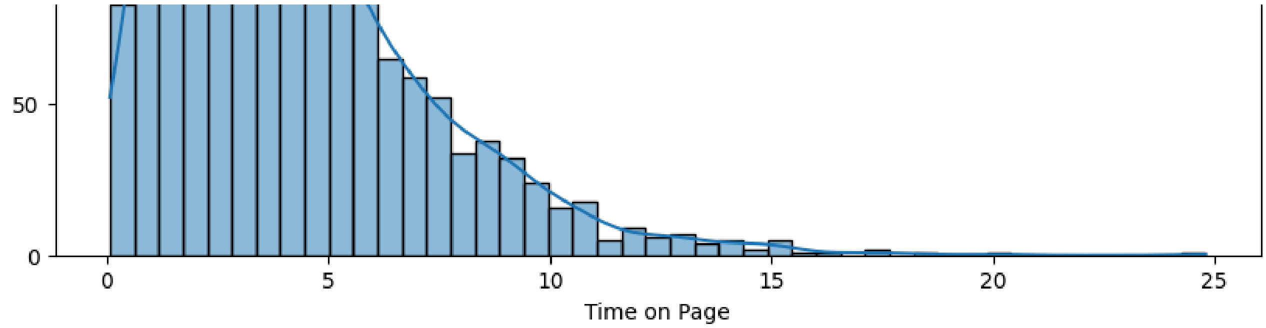

Distribution of Session Duration
Frequency
Session Duration
0
50
100
150
200
250
300
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

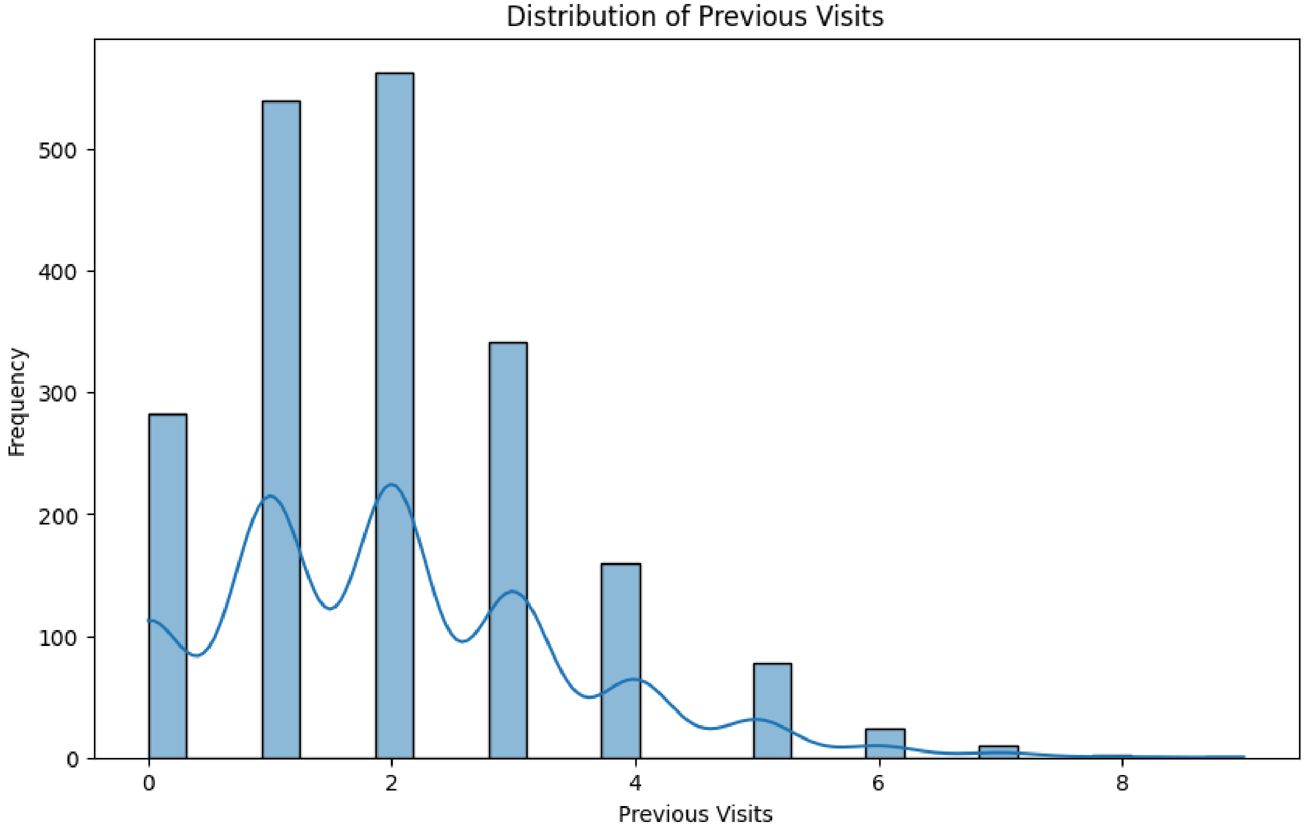

Distribution of Bounce Rate
140
160

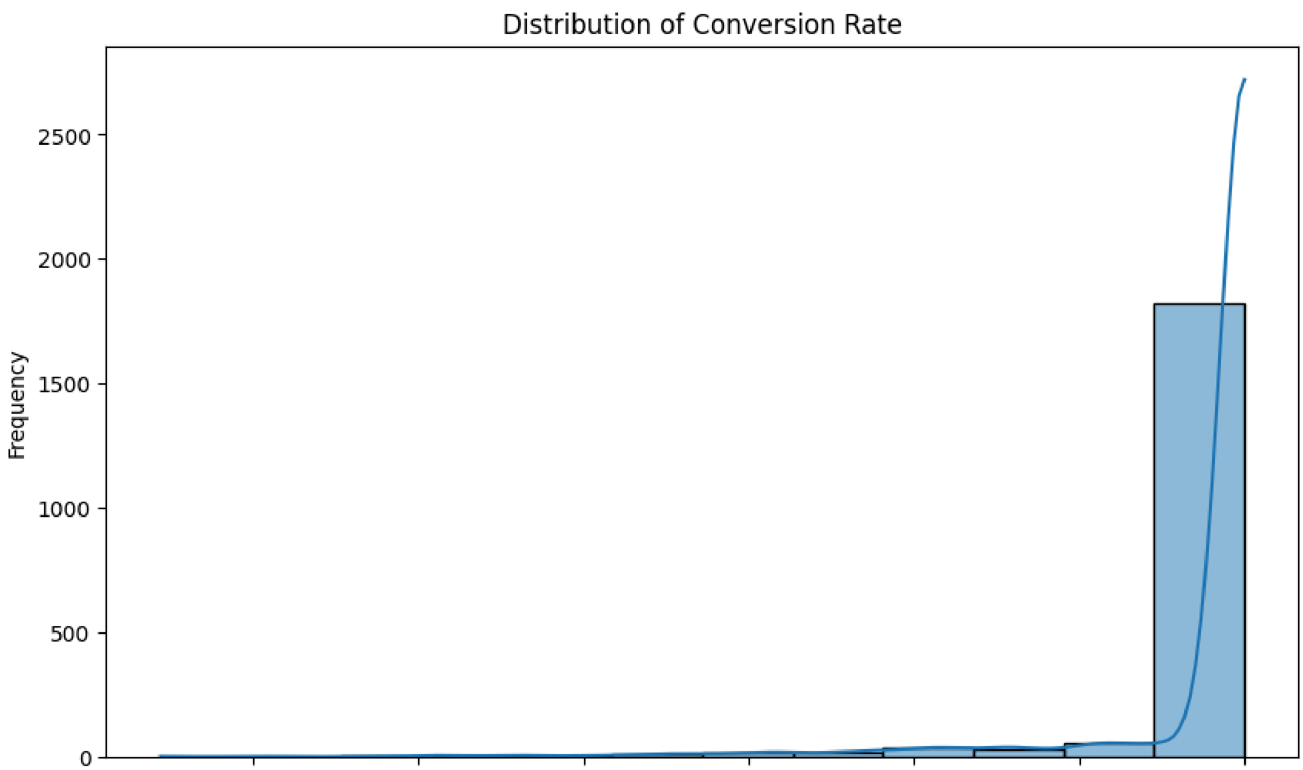

Frequency
Bounce Rate
0
20
40
60
80
100
120
0.0
0.2
0.4
0.6
0.8


Distribution of Traffic Source
Frequency
Traffic Source
Organic
Social
Paid
Direct
Referral
0
200
400
600
800
1000


Distribution of Time on Page
Frequency
100
150
200


50
0
0
5
10
15
20
25
Time on Page


Distribution of Previous Visits
Frequency
500
400
300
200
100
0
0
2
4
6
8
Previous Visits

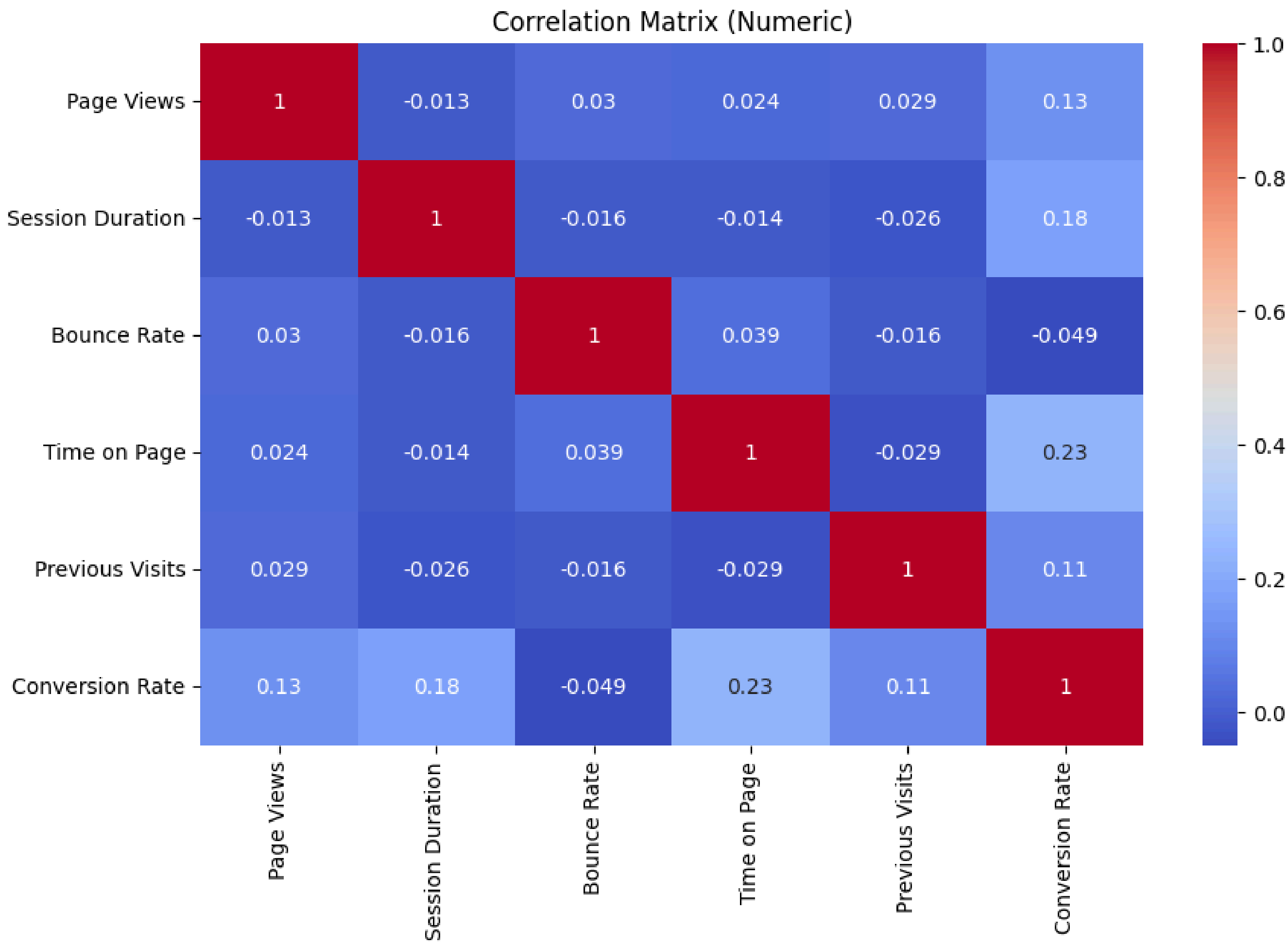

Distribution of Conversion Rate
Frequency
2500
2000
1500
1000
500
0

0.4 0.5 0.6 0.7 0.8 0.9 1.0

Conversion Rate

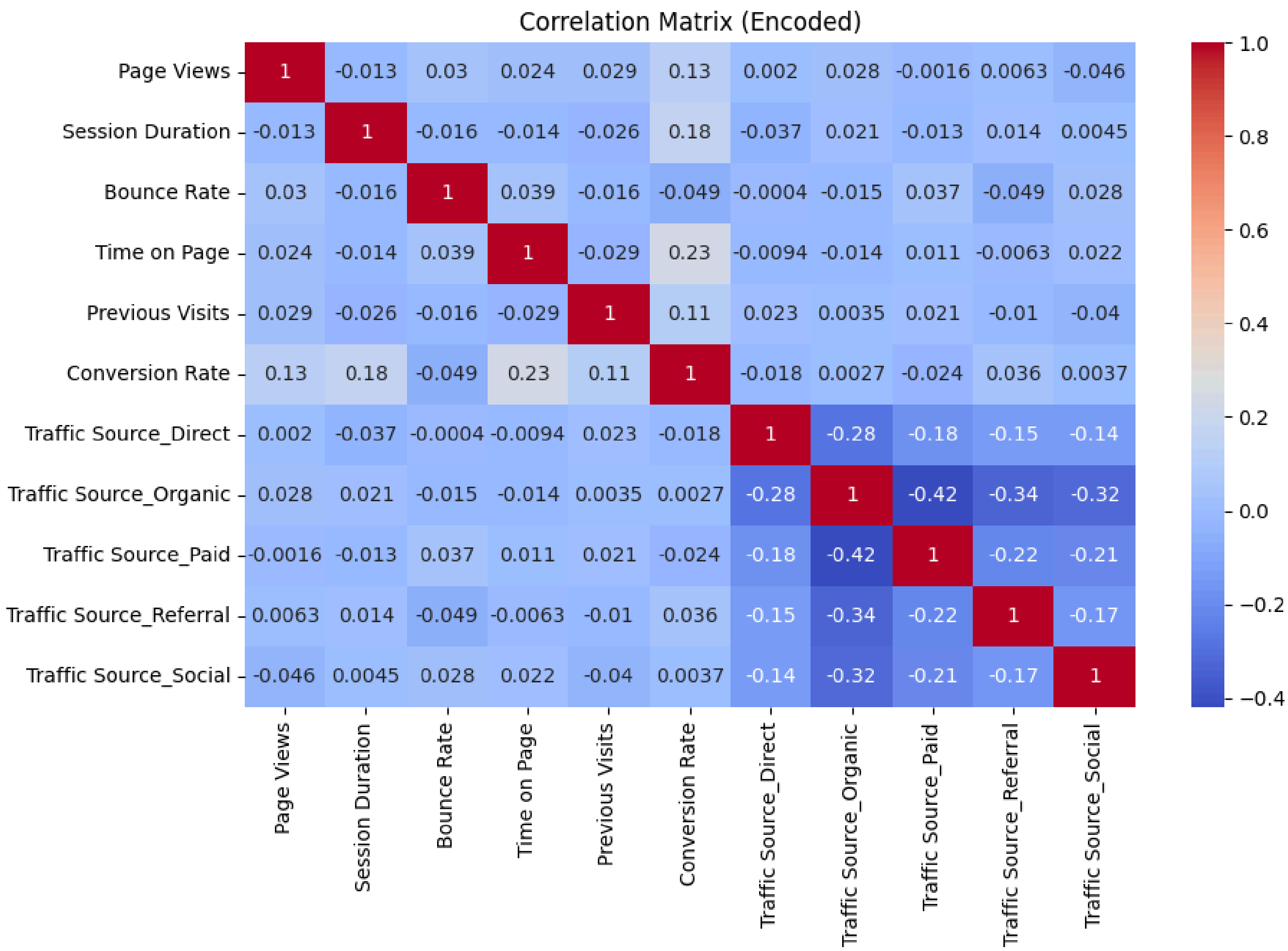
It appears several features have a right-tail skew. The majority of data is clustered near the bottom of the data, but several possible outliers may be changing its shape. Features such as 'Session Duration', 'Previous Visits', 'Time on Page', and others all share this quirk.

```
df_numeric = df.select_dtypes(include=[np.number])

correlation = df_numeric.corr()

plt.figure(figsize=(10,6))
sns.heatmap(data=correlation, cmap='coolwarm', annot=True)
plt.title('Correlation Matrix (Numeric)')
plt.show()
```

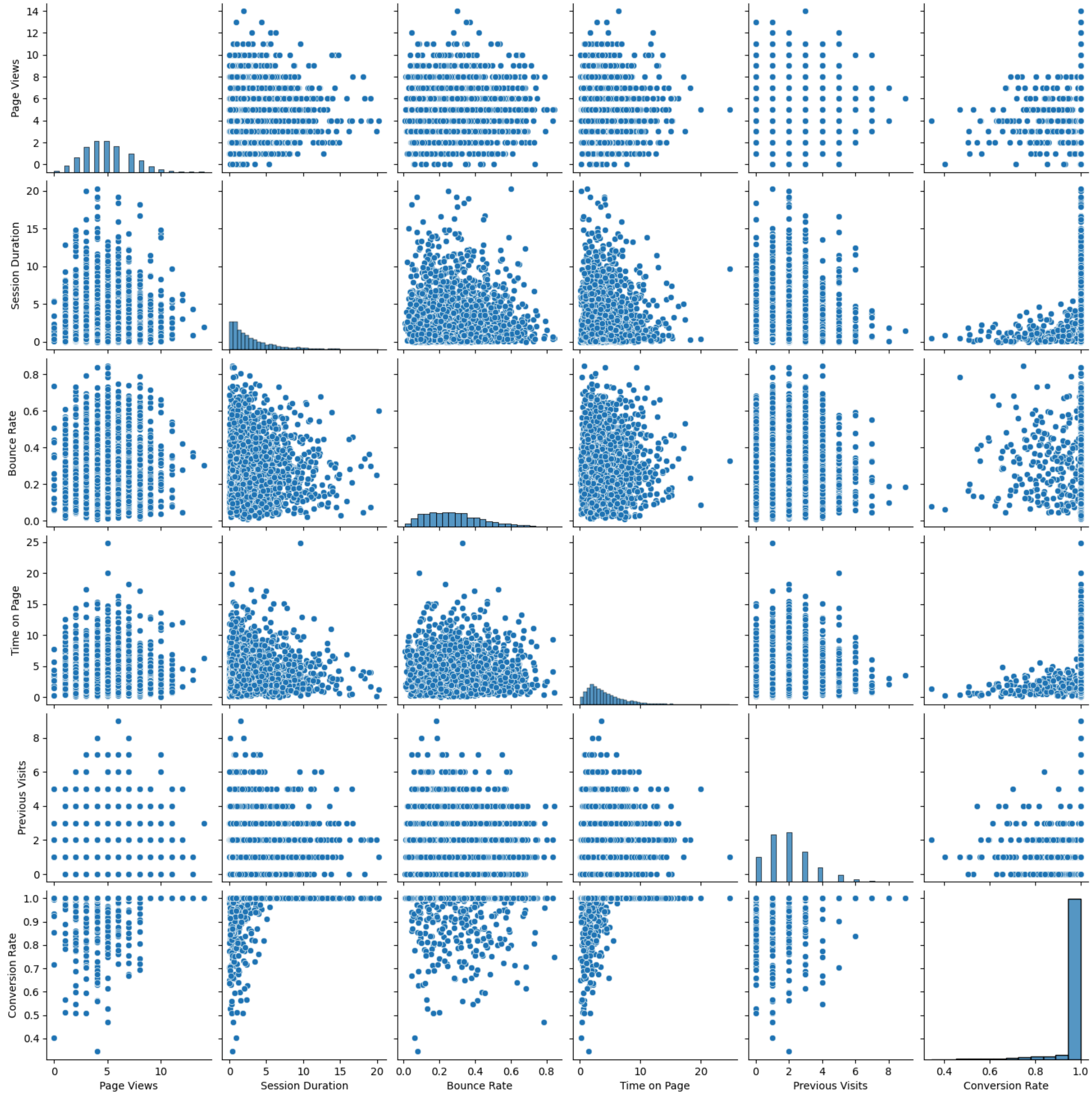



```
# One-hot encoding traffic type for correlation analysis
df_encoded = pd.get_dummies(df, columns=['Traffic Source'])
```

```
correlation_encoded = df_encoded.corr()

plt.figure(figsize=(10,6))
sns.heatmap(data=correlation_encoded, cmap='coolwarm', annot=True)
plt.title('Correlation Matrix (Encoded)')
plt.show()
```




```
# Pairplot for feature analysis
sns.pairplot(df)
plt.show()
```



All features in the dataset appear to have very low correlation with one another. Combined with all of the right-tail skews in the features, I suspect this may be due to the presence of outliers in the data. I will perform z-score and Inter Quartile Range analysis to test this.

```
# Finding outliers through z-score
df_outliers = df.copy()
outliers = []

for col in df_outliers.columns:
  if df_outliers[col].dtype in ['int64', 'float64']:
    mean = df_outliers[col].mean()
    std = df_outliers[col].std()

    z_scores = (df_outliers[col] - mean) / std

# Printing all values with |z-score| > 3
for value in z_scores:
  if abs(value) > 3:
    outliers.append(value)

print(outliers)
```

```
[-3.5454072159846204, -5.431979194201322, -3.0674235923700857, -6.901097028876236, -7.21
```

```
numeric_columns = df.select_dtypes(include=[np.number])
outliers_iqr = find_outliers_iqr(numeric_columns)
print(outliers_iqr)
```

```
[11.05138124  9.63616963  9.68859192 13.58023187 10.02890318 18.3366796
  8.79349225 11.         11.69293807 11.65038153 15.0222197  10.
  9.00440239  9.         11.         15.0497133   9.23837253 14.33851631
  9.84027064 10.30849747 18.23996376  9.          9.          9.
  8.80059289 10.          9.65757692  9.         11.40041688  9.0817998
  9.2274092  11.62537824  9.64423172 10.67944941 10.97709127  9.
 14.26480293  9.79144119 12.26653769  8.89755114  9.04986825  9.12855873
 11.3484583  10.          9.49737922 10.         10.         13.33602859
 11.         11.65479442 10.          9.02713664  9.16915655 16.28439777
 10.9421283  13.54995784 20.0212847   9.72149194 10.75544236 10.
 10.         13.63962071  9.         13.26421013 10.21244597  9.15937315
 10.41547871 10.         14.33777637 12.2619853   9.          9.0695977
  9.05531257  9.         11.34137139 10.38416239 14.49940368 10.52734538
  9.59139401 12.00702784 10.         13.77601687 11.02958591 13.36763409
 12.80147015  9.          9.55999952  9.          8.95568161 10.
 11.82438278 10.52541519 10.45533011 10.         12.72859359  9.40939513
 12.96431643 16.32290146  9.         12.         12.08349527 10.
  9.          9.54736229  9.         11.          9.13028533 19.93267164
 10.62888345 11.62391698 10.28372383 10.73116579 11.43598842 11.
 12.80008785  9.2959347   9.5483859   9.03304007 19.14363617  9.39737406
  9.         15.27774418  9.1816104  10.         13.8486278   9.
  9.         11.          9.          9.03133694 14.55003526 11.
 12.13369056  9.70128077  9.40835794 13.68771409  8.78233322 12.88332128
 11.         10.         11.         10.71673503 11.         10.
 10.26290004  9.          9.17058449 10.          8.86475458  9.2460313
 13.68943989 10.          9.9276564   9.          9.         13.61465526
 10.78251722 10.          8.80277839 14.45754852  9.96589233  9.
  9.          9.          9.         10.43653859 10.          8.96751564
```

```
  9.          9.          9.          9.         10.         11.07541224
 10.         12.40890476  9.3310172   9.13607213  9.         13.88265279
  9.          9.7659436   9.         10.56437726  9.16624747 10.21838447
 11.          9.6449074  15.48708465  9.53525971  9.          8.88604007
  9.21501953 11.51306737  9.          9.59640279 10.29315418 16.67657801
  9.         11.52798876 10.48324902 10.6893614   9.         10.86486752
 18.9563734  12.09657137  9.97781544  9.         10.98585239  8.84950476
 10.70124688 11.81822948 10.11134025  9.58306823 10.39256813 11.34073594
  9.19154493  9.         13.1017075   9.21772402  9.          9.20957212
 11.94223739 20.29051597 10.94155166 11.77041037 10.          9.02040223
 10.87521453 10.40772232  9.48149627 18.16045185  9.33093606 10.4732883
  9.56802055 13.23233754  8.90209302 15.92677174  9.17579809  9.13277539
  9.32370065  9.50834245 15.11562814  9.48255143  9.         10.70188101
  9.29714524  9.57557761 16.55767846 14.31739623 12.03847918 10.12761527
  9.         11.07889148  9.83254018 12.91064307 11.73366226 10.04452973
  9.56051428  9.5669787   9.31248754  9.93669039 10.57907056  8.78230128
 10.92763109  9.4318925   9.         15.10229463 10.6440976  12.
  9.          9.72623619  9.         10.          9.28364265  9.47916263
 14.68715848 17.83503341 10.         17.41481753 10.51224027 10.37711062
 12.5211179  13.28557964 14.          8.90302423  9.63620996  9.27914382
 12.64307642  9.65124136 24.7961822  11.          9.         10.58489758
 13.99005495 11.13254018 10.          9.         14.30578208  9.
 10.         11.05834852 12.72992129 10.77966016  9.74399319 15.0461243
  9.         12.92470262  9.2171389  10.73565577 17.13063475  9.69535459
 11.61178949  8.85497598  9.18913468  9.82909176  8.98928422 10.388859
 11.50898232 12.14598649  9.62006223 12.07878624 10.         10.88130389
 12.39653766 10.         14.63294927 19.1369552   9.         14.10801104
 10.73190312  9.         13.          9.         10.75461815 11.
 12.37674842 13.          8.80489367 10.34502843 11.33579979  9.
```

Based on EDA, it appears that there are no null values in the dataset. However, several features appear to be skewed, and there are multiple outliers in the data. These will need to be removed before analysis for accurate results.

PREPARING THE DATA

```
from scipy import stats
```

```
# Removing outliers
# Computing Z-scores for all numeric columns
z_scores = np.abs(stats.zscore(df.select_dtypes(include=[np.number])))

# Keeping rows where all Z-scores are below the threshold (e.g., 3)
df_no_outliers_z = df[(z_scores < 3).all(axis=1)]
```

```
df_no_outliers_z.shape
```

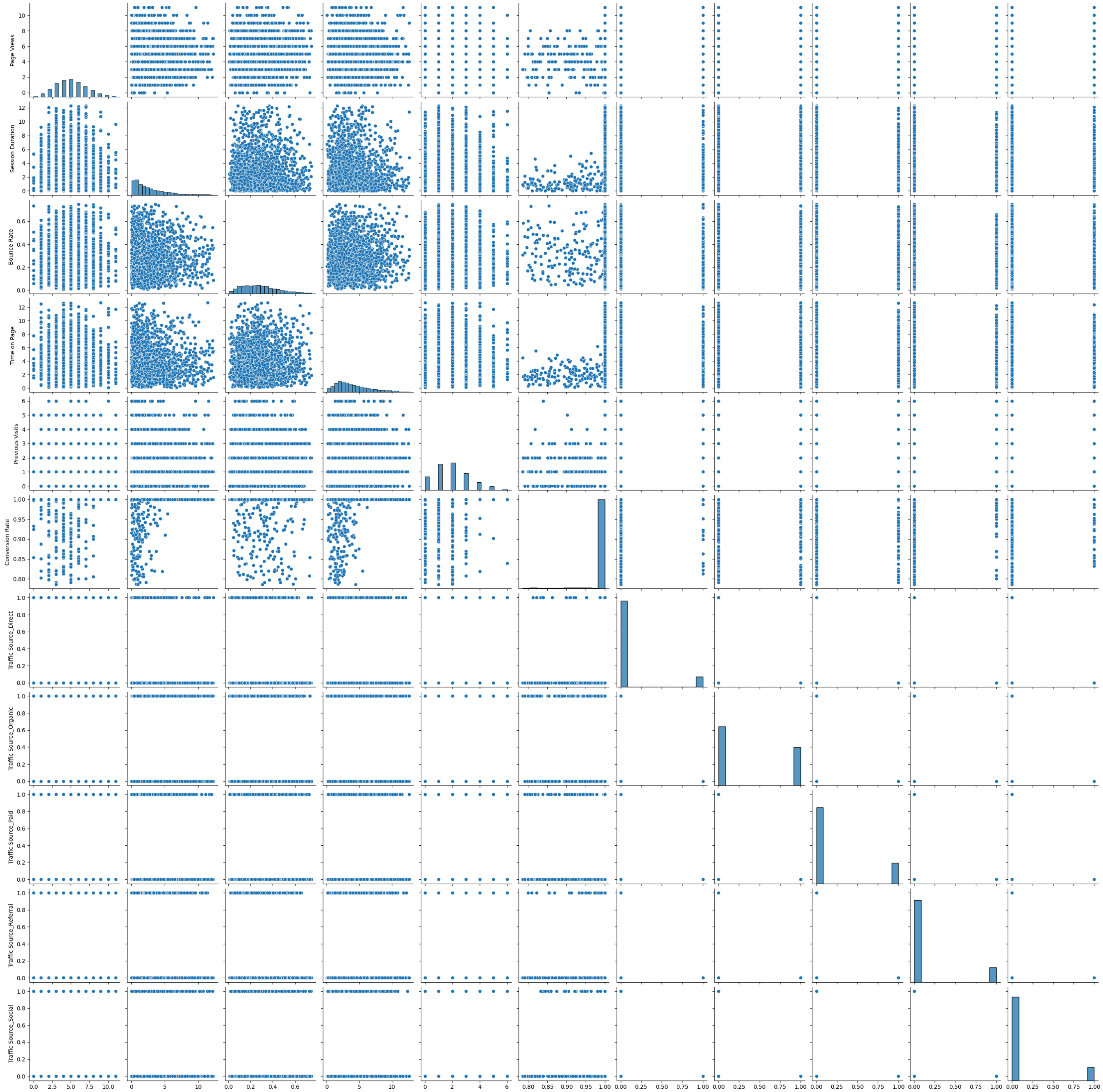
```
(1849, 7)
```

Removing outliers changed the number of rows from 2000 even to 1849, a drop of 151 rows.

```
# One-hot encoding 'Traffic Source' for linear regression
df_encoded = pd.get_dummies(df_no_outliers_z, columns=['Traffic Source'])
df_encoded.head()
```

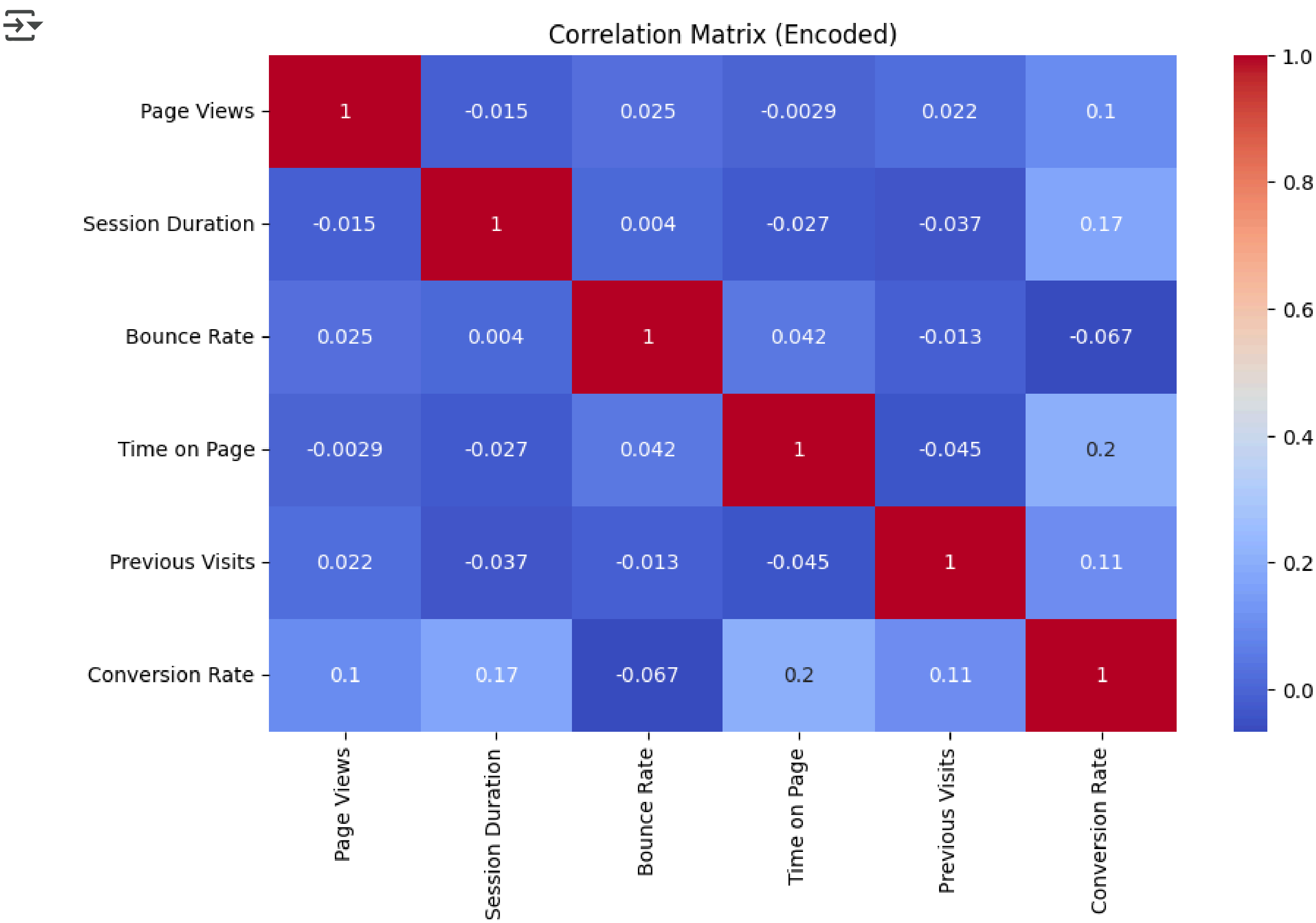
	Page Views	Session Duration	Bounce Rate	Time on Page	Previous Visits	Conversion Rate	Traffic Source_Direct	Traf Source_Orga
0	5	11.051381	0.230652	3.890460	3	1.0	False	T
1	4	3.429316	0.391001	8.478174	0	1.0	False	Fa
2	4	1.621052	0.397986	9.636170	2	1.0	False	T
3	5	3.629279	0.180458	2.071925	3	1.0	False	T
4	5	4.235843	0.291541	1.960654	5	1.0	False	Fa

```
# Pairplot for feature analysis w/o outliers
sns.pairplot(df_encoded)
plt.show()
```



The pairplot without outliers shows much more interesting information. Page views and bounce rate become much less skewed, and the connections between features and "conversion rate" become more pronounced - particularly "session duration", "bounce rate", and "time on page".

```
numeric_columns = df_encoded.select_dtypes(include=[np.number])
correlation_encoded = numeric_columns.corr()

plt.figure(figsize=(10,6))
sns.heatmap(data=correlation_encoded, cmap='coolwarm', annot=True)
plt.title('Correlation Matrix (Encoded)')
plt.show()
```



With outliers removed, some values gain more of a correlation:

- Session Duration has a (albeit very low) positive correlation with Conversion Rate.
- Time on Page also has a (albeit very low) positive correlation with Conversion Rate.

MACHINE LEARNING