Search Queries Anomaly Detection using python

In [4]:
```
import pandas as pd
from collections import Counter
import re
import plotly.express as px
import plotly.io as pio
pio.templates.default = "plotly_white"
```

In [5]:
```
queries_df = pd.read_csv(r"C:\Users\JANHAVI\Desktop\Dataset\Queries.csv")
print(queries_df.head())
```

```
                              Top queries  Clicks  Impressions     CTR  \
0               number guessing game python    5223        14578  35.83%
1                      thecleverprogrammer    2809         3456  81.28%
2          python projects with source code    2077        73380   2.83%
3  classification report in machine learning    2012         4959  40.57%
4                     the clever programmer    1931         2528  76.38%

   Position
0      1.61
1      1.02
2      5.94
3      1.28
4      1.09
```

In [6]:
```
import warnings
warnings.filterwarnings("ignore")
```

Exploratory Data Analysis

In [7]:
```
print(queries_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Top queries  1000 non-null   object 
 1   Clicks       1000 non-null   int64  
 2   Impressions  1000 non-null   int64  
 3   CTR          1000 non-null   object 
 4   Position     1000 non-null   float64
dtypes: float64(1), int64(2), object(2)
memory usage: 39.2+ KB
None
```

In [16]:
```
queries_df['CTR'] = queries_df['CTR'].astype(str)
```

In [17]:
```
# Function to clean and split the queries into words
def clean_and_split(query):
    words = re.findall(r'\b[a-zA-Z]+\b', query.lower())
    return words

# Split each query into words and count the frequency of each word
```

```
word_counts = Counter()
for query in queries_df['Top queries']:
    word_counts.update(clean_and_split(query))

word_freq_df = pd.DataFrame(word_counts.most_common(20), columns=['Word', 'Frequenc

# Plotting the word frequencies
fig = px.bar(word_freq_df, x='Word', y='Frequency', title='Top 20 Most Common Words
fig.show()
```

In [18]:
```
# Top queries by Clicks and Impressions
top_queries_clicks_vis = queries_df.nlargest(10, 'Clicks')[['Top queries', 'Clicks'
top_queries_impressions_vis = queries_df.nlargest(10, 'Impressions')[['Top queries'

# Plotting
fig_clicks = px.bar(top_queries_clicks_vis, x='Top queries', y='Clicks', title='Top
fig_impressions = px.bar(top_queries_impressions_vis, x='Top queries', y='Impressio
fig_clicks.show()
fig_impressions.show()
```

In [23]:
```
import numpy as np
# Clean CTR column before using nlargest/nsmallest
queries_df['CTR'] = (queries_df['CTR'].astype(str).str.replace('%', '', regex=False

# Queries with highest and lowest CTR
top_ctr_vis = queries_df.nlargest(10, 'CTR')[['Top queries', 'CTR']]
bottom_ctr_vis = queries_df.nsmallest(10, 'CTR')[['Top queries', 'CTR']]

# Plotting
fig_top_ctr = px.bar(top_ctr_vis, x='Top queries', y='CTR', title='Top Queries by C
fig_bottom_ctr = px.bar(bottom_ctr_vis, x='Top queries', y='CTR', title='Bottom Que

fig_top_ctr.show()
fig_bottom_ctr.show()
```

In [24]:
```
# Correlation matrix visualization
correlation_matrix = queries_df[['Clicks', 'Impressions', 'CTR', 'Position']].corr(
fig_corr = px.imshow(correlation_matrix, text_auto=True, title='Correlation Matrix'
fig_corr.show()
```

Detecting Anomalies in Search Queries

In [25]:
```
from sklearn.ensemble import IsolationForest

# Selecting relevant features
features = queries_df[['Clicks', 'Impressions', 'CTR', 'Position']]

# Initializing Isolation Forest
iso_forest = IsolationForest(n_estimators=100, contamination=0.01)  # contaminatior

# Fitting the model
iso_forest.fit(features)

# Predicting anomalies
queries_df['anomaly'] = iso_forest.predict(features)

# Filtering out the anomalies
anomalies = queries_df[queries_df['anomaly'] == -1]
```

In [26]:
```
print(anomalies[['Top queries', 'Clicks', 'Impressions', 'CTR', 'Position']])
```

	Top queries	Clicks	Impressions	CTR	Position
0	number guessing game python	5223	14578	0.3583	1.61
1	thecleverprogrammer	2809	3456	0.8128	1.02
2	python projects with source code	2077	73380	0.0283	5.94
4	the clever programmer	1931	2528	0.7638	1.09
15	rock paper scissors python	1111	35824	0.0310	7.19
21	classification report	933	39896	0.0234	7.53
34	machine learning roadmap	708	42715	0.0166	8.97
82	r2 score	367	56322	0.0065	9.33
167	text to handwriting	222	11283	0.0197	28.52
929	python turtle	52	18228	0.0029	18.75

In [ ]:

In [ ]:

In [ ]: