AICP Internship Task

Search Queries Anomaly Detection is a technique to identify unusual or unexpected patterns in search query data. Below is the process we can follow for the task of Search Queries Anomaly Detection:

1. Gather historical search query data from the source, such as a search engine or a website's search functionality.
2. Conduct an initial analysis to understand the distribution of search queries, their frequency, and any noticeable patterns or trends.
3. Create relevant features or attributes from the search query data that can aid in anomaly detection.
4. Choose an appropriate anomaly detection algorithm. Common methods include statistical approaches like Z-score analysis and machine learning algorithms like Isolation Forests or One-Class SVM.
5. Train the selected model on the prepared data.
6. Apply the trained model to the search query data to identify anomalies or outliers

Find the Dataset "**Queries.csv**".

The dataset we have contains search queries that lead users to a specific website, along with associated metrics. The columns in this dataset are:

- **Top Queries:** The actual search terms used by users.
- **Clicks:** The number of times users clicked on the website after using the query.
- **Impressions:** The number of times the website appeared in search results for the query.
- **CTR (Click Through Rate):** The ratio of clicks to impressions, indicating the effectiveness of the query in leading users to the website.
- **Position:** The average ranking of the website in search results for the query.

The problem at hand is to utilize the available dataset to detect anomalies in search queries — queries that perform significantly differently from the majority. The goal is to identify queries that are either underperforming or overperforming in terms of clicks, impressions, CTR, and search position.

Import Necessary libraries like:

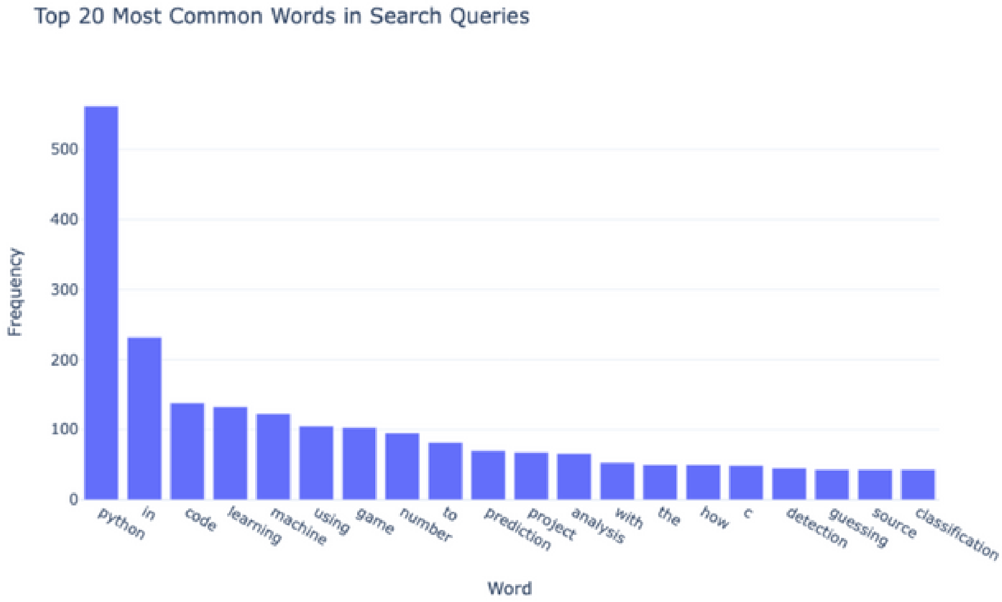
```
import pandas as pd
from collections import Counter
import re
import plotly.express as px
import plotly.io as pio
pio.templates.default = "plotly_white"
```

Q.1: Import data and check null values, check column info and the descriptive statistics of the data.

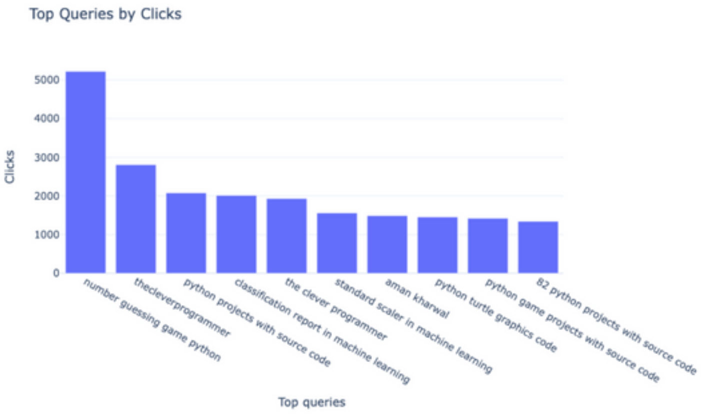
Q.2: Now convert the CTR column from a percentage string to a float

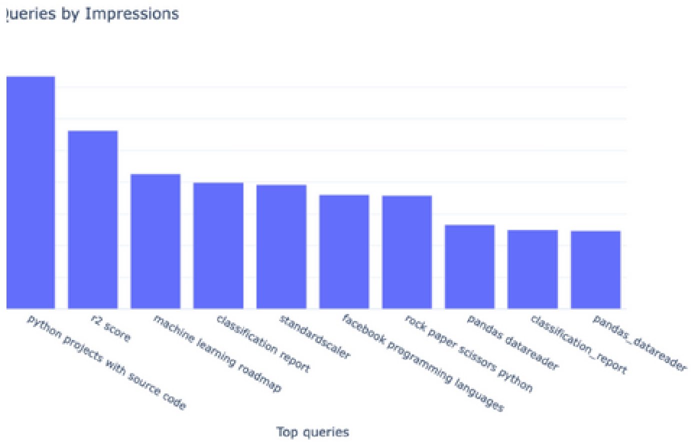
Q.3: Now analyze common words in each search query in the following manner:

- Create a function to clean and split the queries into words.
- Split each query into words and count the frequency of each word.
- Plot the word frequencies

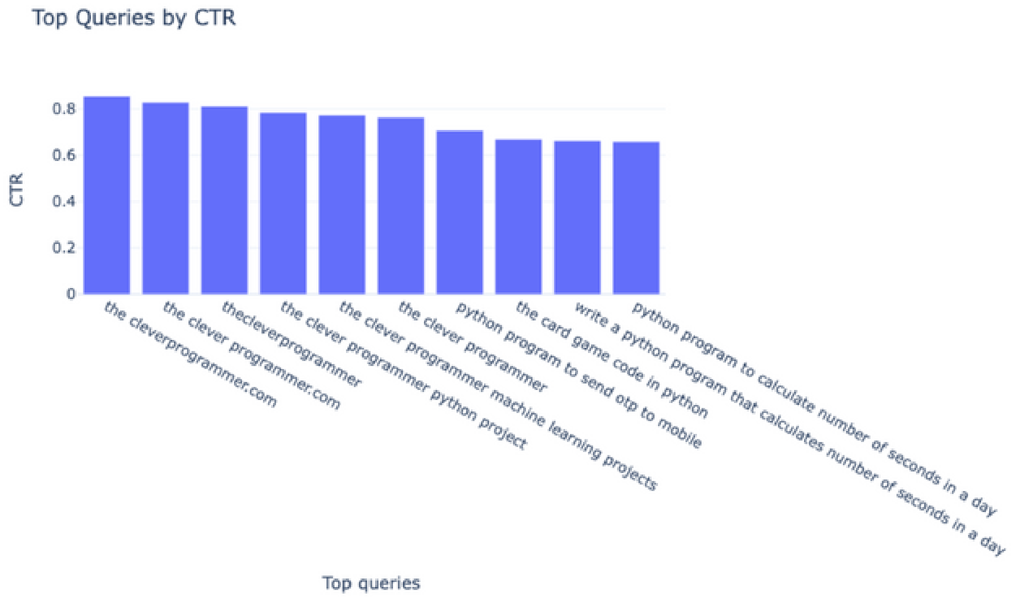


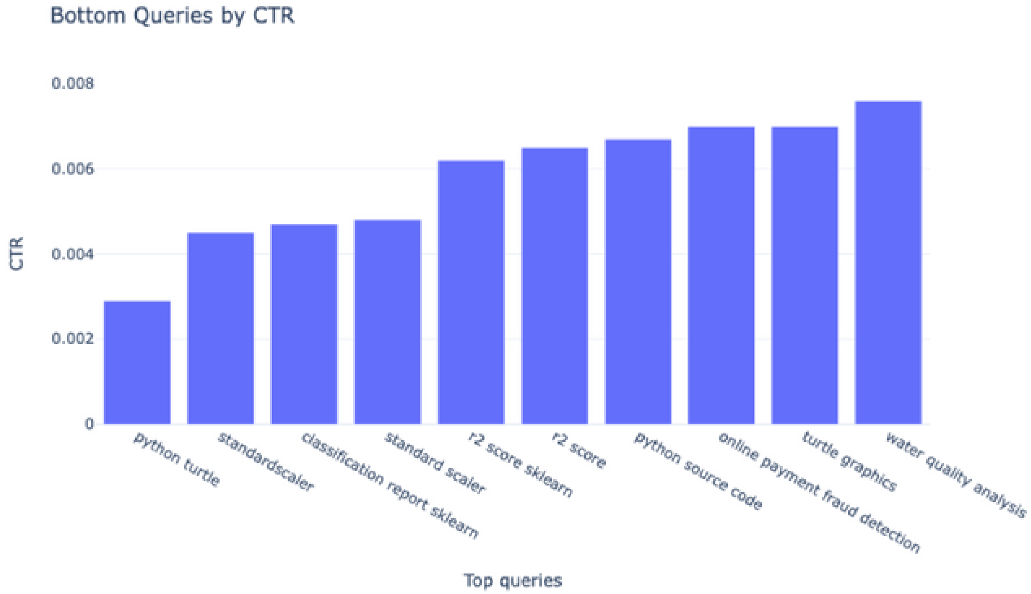
Q.4: Now have a look at the top queries by clicks and impressions





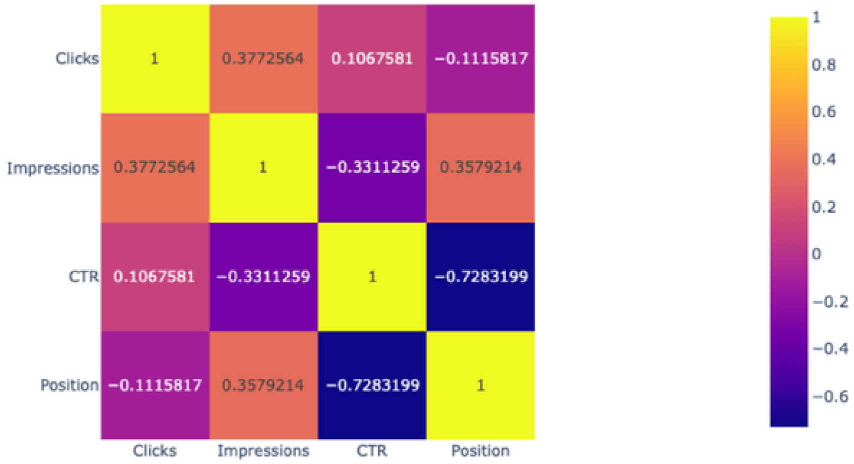
Q.5: Now analyze the queries with the highest and lowest CTRs





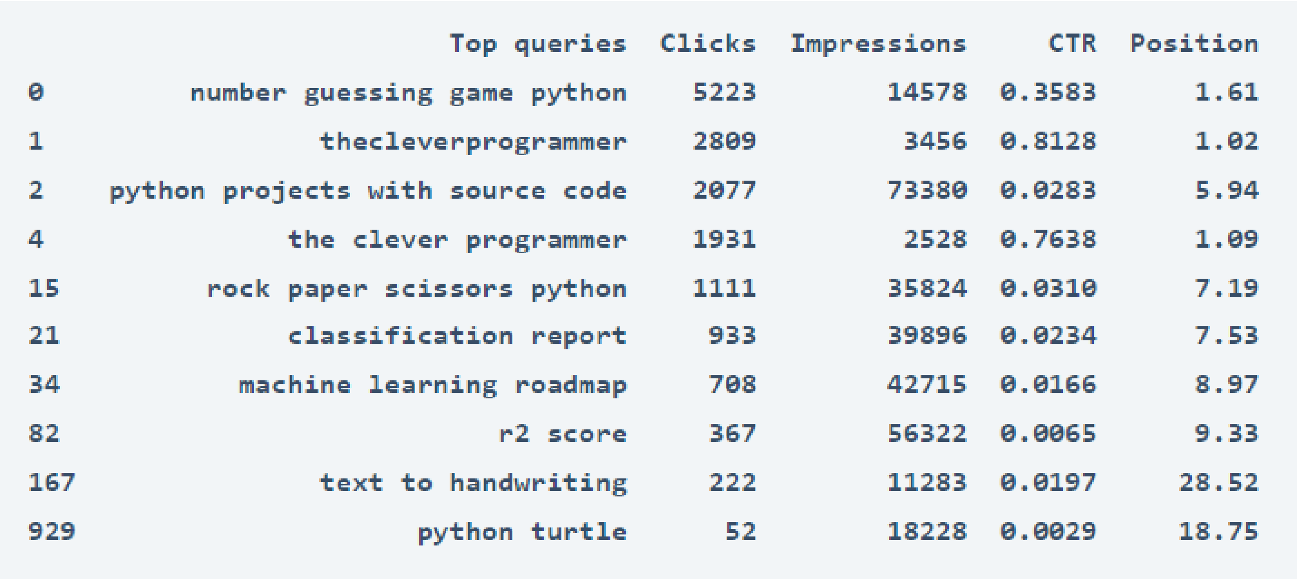
Q.6: Now check the correlation between different metrics. Also explain your observation from the correlation matrix

Correlation Matrix



Q.7: Now, detect anomalies in search queries. You can use various techniques for anomaly detection. A simple and effective method is the Isolation Forest algorithm, which works well with different data distributions and is efficient with large datasets.

Show results like this



```
                      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
```