



Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Фізико-технічний інститут

Лабораторна робота №1
з дисципліни
«Web - аналітика»

Виконав:
студент групи ФБ-31мп
Щур Павло
Перевірив:
Ткач В. М.

Київ-2024

Посилання на GitHub: <https://github.com/ShchurPavlo/web-analytics-2024/tree/main/lab1>

Завдання

1. На основі будь-якого access.log сформувати датасет, що надав би інформацію про користувачів веб-ресурсу, а потім виконати наступні кроки:

- a. Визначити кількість користувачів за днями
- b. Ранжувати користувачів за User-Agent
- c. Ранжувати користувачів за операційними системами
- d. Ранжувати користувачів за країною запиту
- e. Виокремити пошукових ботів
- f. Детектувати аномалії (якщо такі є)

Виконання:

1) В якості піддослідного датасету використаємо

<https://www.kaggle.com/datasets/adevvenugopal/webserverlogs10k/code> .

Розпарсимо його на окремі параметри - для цього реалізуємо дві функції, перша з яких зчитує порядково дані з файлу датасету та передає їх у функцію-парсер:

```
def Get_data(path):  
    data=[]  
    with open(path, 'r') as file:  
        lines = file.readlines()  
        for line in lines:  
            tmp=Parse_line(line)  
            if tmp is not None:  
                data.append(tmp)  
    df = pandas.DataFrame(data)  
    return df
```

В свою чергу функція парсер за допомогою регулярного виразу розбиває дані на окремі параметри. В контексті майбутнього завдання було виокремлено наступні параметри:

- IP-адреса
- Країна (визначається за допомогою geoip2.database на основі IP адреси)
- Дата підключення
- Час підключення
- Розмір переданих даних
- URL запиту
- User-агент підключення
- Браузер підключення
- Операційна система

```

def Parse_line(line):
    pattern = re.compile(r'(\d+), '
                        r'"((?:\d{1,3}\.){3}\d{1,3}) - - \[(\d{2})/[A-Za-z]{3}/\d{4}:\d{2}:\d{2}:'
    match = pattern.match(line)
    if match:
        parameters = match.groups()
        user_agents_obj=parse(parameters[7])
        datetime_obj = datetime.datetime.strptime(parameters[2], _format: '%d/%b/%Y:%H:%M:%S %z')
        return {
            "IP": parameters[1],
            "Country": Get_country(parameters[1]),
            "Date": datetime_obj.date(),
            "Time": datetime_obj.time(),
            "Request type": parameters[3],
            "Request code": parameters[4],
            "Size": int(parameters[5]),
            "Request url": parameters[6],
            "User_agents": parameters[7],
            "Browser": user_agents_obj.browser,
            "OS": user_agents_obj.os.family
        }
    else:
        return None

```

2) Для обрахунку кількості з'єднань по днях реалізуємо наступну функцію:

```

50 def Calc_unic_users(data):
51     print("Users by days:")
52     result = data.groupby('Date')['IP'].nunique().sort_values(ascending=False)
53     print(result)

```

Результат роботи:

```

Users by days:
Date
2019-01-26    1231
2019-01-23    1223
2019-01-22    1113
2019-01-25    1110
2019-01-24    1028

```

3) Для визначення операційних систем з яких здійснюється підключення реалізуємо наступну функцію:

```

55 def Calc_unic_os(data):
56     print("Users by OS:")
57     result = data.groupby('OS')['IP'].nunique().sort_values(ascending=False)
58     print(result)

```

Результат роботи:

```
Users by OS:  
OS  
Windows          2825  
Android          2215  
iOS              441  
Mac OS X         33  
Linux            27  
Other            10  
Windows Phone    4  
Ubuntu           3  
BlackBerry OS    1  
Tizen            1  
Name: IP, dtype: int64
```

4) Ранжуємо користувачів за країною запиту, для цього також використаємо окрему функцію:

```
65 def Calc_uniq_contry(data):  
66     print("Countries:")  
67     result = data.groupby('Country')['IP'].nunique().sort_values(ascending=False)  
68     print(result)
```

Результат роботи :

```
Countries:  
Country  
Iran          4796  
United States  221  
Germany       171  
The Netherlands  54  
Finland       48  
France        45  
United Kingdom 45  
Syria         20  
Australia     11  
Canada        10  
Türkiye       8  
Japan         7  
Bulgaria      5  
Singapore     5
```

5) В піддослідному датасеті всі User-agents є унікальними, тому реалізуємо ранжування підключень за браузером та його версією:

```
60 def Calc_uniq_browser(data):
61     print("Unique browsers:")
62     result = data.groupby('Browser')['IP'].nunique().sort_values(ascending=False)
63     print(result)
```

Результат роботи:

```
Browser
(Chrome, (71, 0, 3578), 71.0.3578)      1585
(Firefox, (64, 0), 64.0)                778
(Chrome Mobile, (71, 0, 3578), 71.0.3578)  568
(Samsung Internet, (8, 2), 8.2)          214
(Mobile Safari, (12, 0), 12.0)           209
...
(Google, (64, 0, 223374052), 64.0.223374052)  1
(Google, (63, 0, 221691834), 63.0.221691834)  1
(Google, (62, 1, 220348572), 62.1.220348572)  1
(Google, (60, 2, 216743813), 60.2.216743813)  1
(Yandex Browser, (18, 11, 1), 18.11.1)       1
```

6) Для виокремлення пошукових ботів сформуємо список типових ботів та в окремій функції реалізуємо виокремлення записів та їх підрахунок:

```
71 def Show_unique_bots(data):
72     bots = ['Googlebot', 'Bingbot', 'Yahoo! Slurp', 'DuckDuckBot', 'Baiduspider']
73     data['Bot'] = data['User_agents'].apply(lambda x: next(
74         (bot for bot in bots if bot in x), None))
75     unique_bots = data.groupby('Bot')['IP'].nunique()
76     print(unique_bots)
```

В результаті отримаємо 3 записи від Googlebot (їх можна вважати аномальними):

```
Bot
Googlebot    3
```

7) Для детекції аномалій будемо обчислювати Z-score для значення поля Size. Значення Z-score - це міра відхилення вимірюваної величини (у нашому випадку Size) від середнього значення, виражена в одиницях стандартного відхилення. Аномалії – це точки, модуль оцінки Z-score яких більше 3.

```
81 def Detect_anomalies(data):
82
83     mean_val = data['Size'].mean()
84     std_dev = data['Size'].std()
85     data['Size_Z_Score'] = data['Size'].apply(lambda x: Calculate_z_score(x, mean_val, std_dev))
86
87     print(data['Size_Z_Score'])
88     df_zscore = data['Size_Z_Score']
89
90     anomalies = data[data['Size_Z_Score'] > 3]
91     print(f"Found {len(anomalies)} anomalies")
92
93     # Plot anomalies
94     plt.figure(figsize=(10, 5))
95     plt.plot(*args: data.index, data['Size'], label='Size', color="blue", alpha=0.25)
96     plt.scatter(anomalies.index, anomalies['Size'], color="green", label="Anomalies", marker="*")
97     plt.title(f"Anomalies")
98     plt.ylabel('Size')
99     plt.grid(True)
100    plt.tight_layout()
101    plt.legend()
102    plt.show()
```

В результаті було знайдено 156 аномальних результати:

```
Name: IP, dtype: int64
0      2.901352
1     -0.320361
2     -0.338772
3     -0.364468
4     -0.221022
...
6303    0.994249
6304   -0.454227
6305   -0.397448
6306    0.404551
6307   -0.435816
Name: Size_Z_Score, Length: 6308, dtype: float64
Found 156 anomalies
```

Візуалізуємо отримані аномалії:

