Міністерство освіти і науки України

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Фізико-технічний інститут

WEB-АНАЛІТИКА

Лабораторна робота

Робота №1

Виконав студент гр. ФБ-31мп:

Шевченко С. Ш.

Київ – 2024

[Репозиторій](https://github.com/dedkuzmich/web-analytics-2024) Github з кодом.

# Завдання:

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 завдань:

В якості вхідних даних візьмемо файл access.log та БД country.mmdb:

[Лог файл](https://github.com/elastic/examples/blob/master/Common%20Data%20Formats/apache_logs/apache_logs) Apache access.log.

[БД](https://ipinfo.io/products/free-ip-database) IP-адрес country.mmdb:



## Створення датасету з файлу access.log

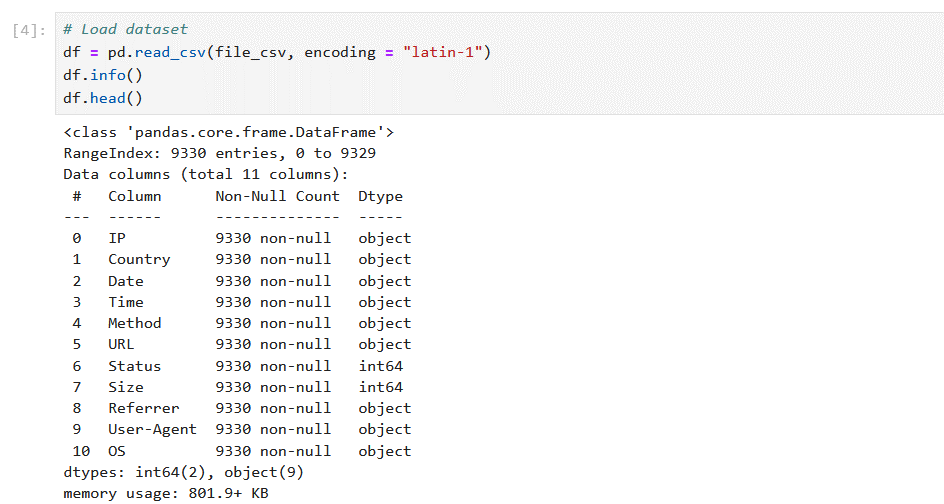
За допомогою regex парсимо кожний рядок файлу access.log, щоб виокремити кожне поле HTTP-запиту:

1. Країну, з якої надійнов запит, можемо дізнатися, використовуючи БД IPinfo (файл country.mmdb). Це можливо, тому що кожній країні встановлюєтсья певний діапазон IP-адрес.
2. ОС клієнта можна дізнатися, розпарсивши User-Agent.

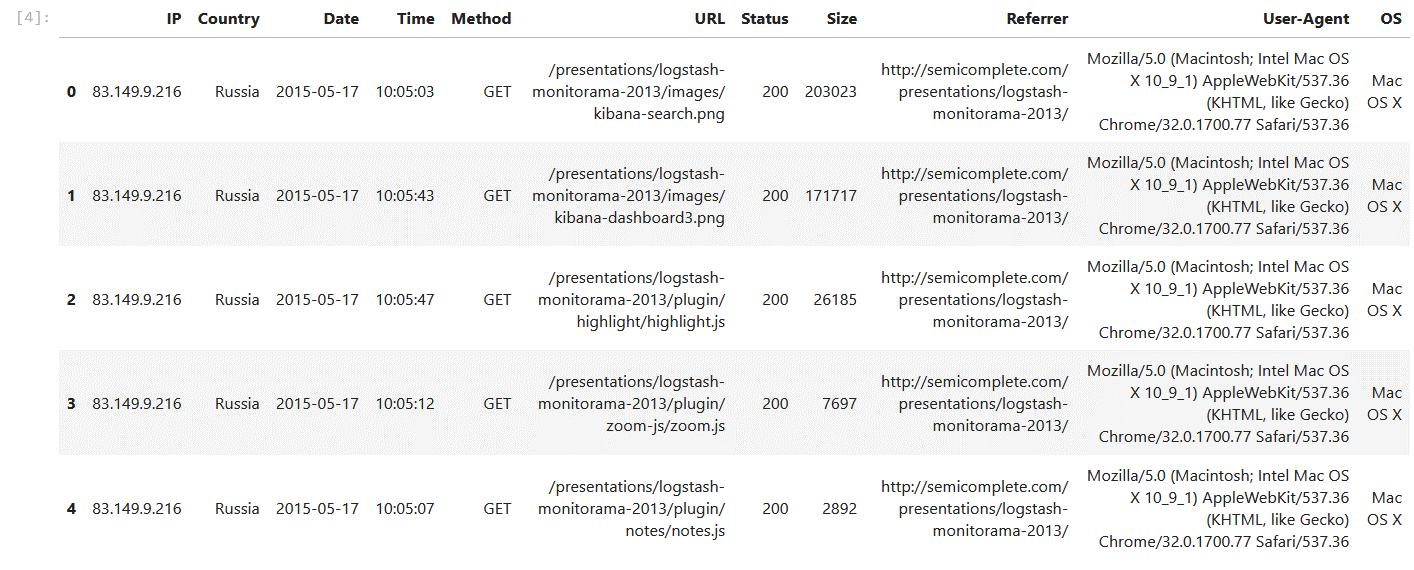
Після обробки всіх рядків логу, об'єднуємо їх в датафрейм pandas і зберігаємо як датасет access.csv.

# Form dataset from Apache access.log  
file\_log = "input/access.log" # Source: https://github.com/elastic/examples/tree/master/Common%20Data%20Formats/apache\_logs  
file\_csv = "input/access.csv"  
file\_ipinfo = "input/country.mmdb" # Source: https://ipinfo.io/products/free-ip-database  
list\_log = []  
  
# Parse access.log with regex  
pattern = r'(?P<ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}) (?P<client>.\*?) (?P<user>.\*?) \[(?P<datetime>.\*?)\] "(?P<method>\w+) (?P<url>.\*?) (?P<protocol>HTTP/1.\d)" (?P<status>\d+) (?P<size>\d+) "(?P<referrer>.\*?)" "(?P<agent>.\*?)"'  
regex = compile(pattern)  
reader = maxminddb.open\_database(file\_ipinfo)  
*with* open(file\_log, "r") *as* fin:  
 *for* line *in* fin:  
 match = regex.match(line)  
 *if not* match: # If cannot parse the line  
 *continue* # Get country by IP  
 ip = match.group("ip")  
 country = reader.get(ip)["country\_name"]  
  
 # Get time/date  
 datetime\_str = match.group("datetime")  
 datetime\_obj = dt.strptime(datetime\_str, '%d/%b/%Y:%H:%M:%S %z')  
 date = datetime\_obj.date()  
 time = datetime\_obj.time()  
  
 # Retrieve OS from User-Agent  
 agent = match.group("agent")  
 os = parse(agent).os.family  
  
 list\_log.append({  
 "IP": ip,  
 "Country": country,  
  
 # "Client": match.group("client"),  
 # "User": match.group("user"),  
  
 "Date": date,  
 "Time": time,  
  
 "Method": match.group("method"),  
 "URL": match.group("url"),  
 # "Protocol": match.group("protocol"),  
  
 "Status": match.group("status"),  
 "Size": match.group("size"),  
  
 "Referrer": match.group("referrer"),  
 "User-Agent": agent,  
 "OS": os  
 })  
  
# Save parsed access.log as CSV  
df = pd.DataFrame(list\_log)  
df.to\_csv(file\_csv, index = *False*)  
print(f"Number of entries: {len(df)}")

Отриманий датасет містить 11 фіч (стовпців):



Перші 5 рядків:



## a. Визначити кількість користувачів за днями

Згрупуємо IP-адреси за днями та підрахуємо їх к-ість за кожен день:

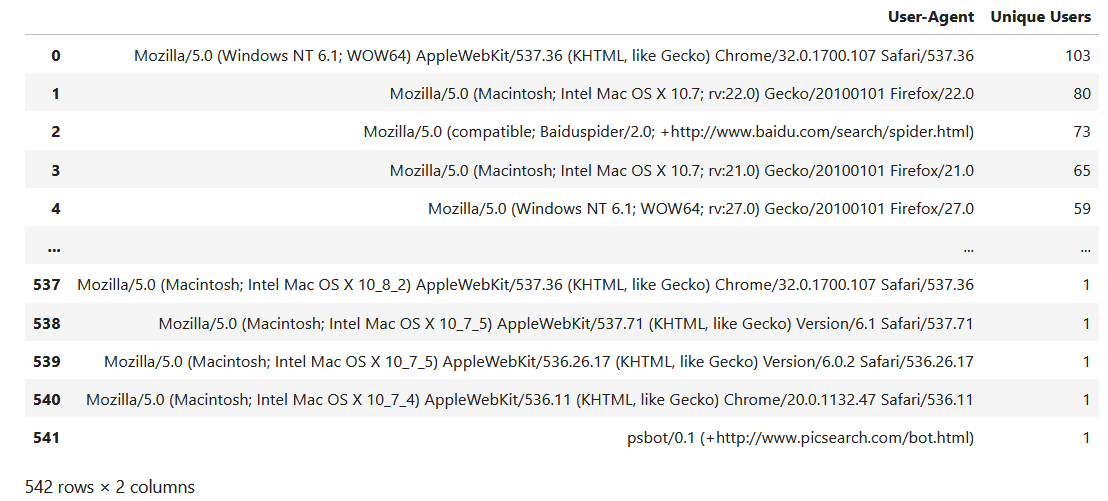
# A) Count users by day  
users\_by\_day = df.groupby("Date")["IP"].nunique()  
users\_by\_day = users\_by\_day.reset\_index().rename(columns = {"IP": "Unique Users"})  
users\_by\_day.to\_csv("output/A\_users\_by\_day.csv", index = *False*)  
display(users\_by\_day)



## b. Ранжувати користувачів за User-Agent

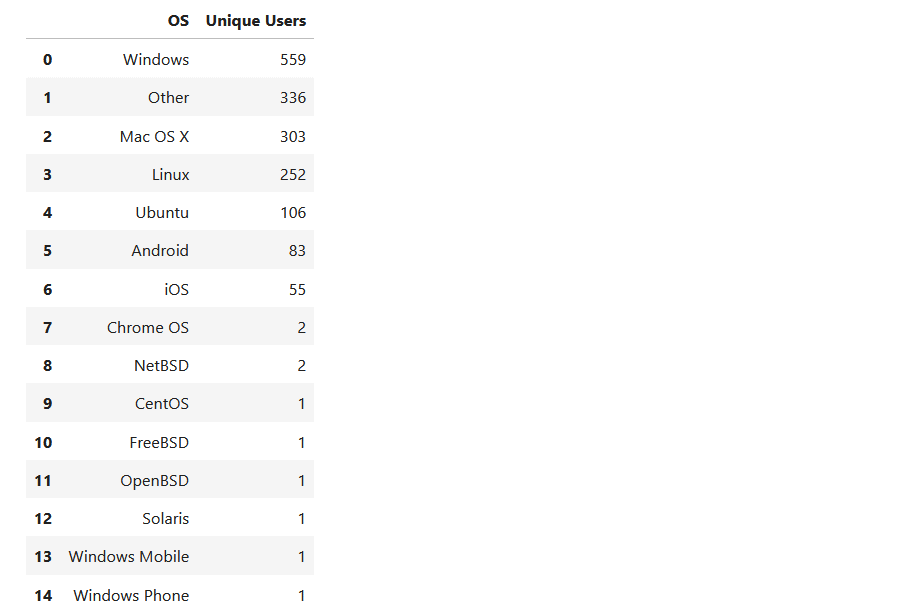
Згрупуємо IP-адреси за User-Agent, посортуємо (порядок сортування - спадаючий):

# B) Sort user agents by users  
top\_user\_agents = df.groupby("User-Agent")["IP"].nunique().sort\_values(ascending = *False*)  
top\_user\_agents = top\_user\_agents.reset\_index().rename(columns = {"IP": "Unique Users"})  
top\_user\_agents.to\_csv("output/B\_top\_user\_agents.csv", index = *False*)  
display(top\_user\_agents)



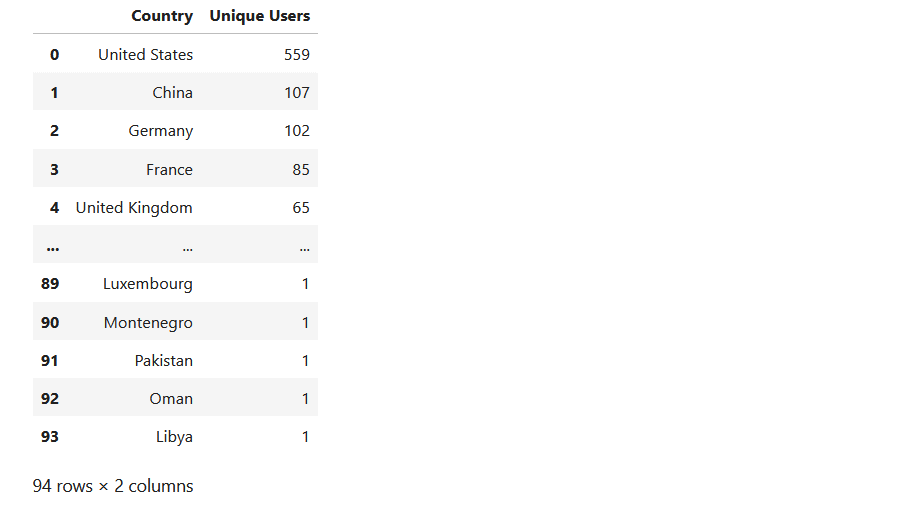
## c. Ранжувати користувачів за операційними системами

# C) Sort OS by users  
top\_os = df.groupby("OS")["IP"].nunique().sort\_values(ascending = *False*)  
top\_os = top\_os.reset\_index().rename(columns = {"IP": "Unique Users"})  
top\_os.to\_csv("output/C\_top\_os.csv", index = *False*)  
display(top\_os)



## d. Ранжувати користувачів за країною запиту

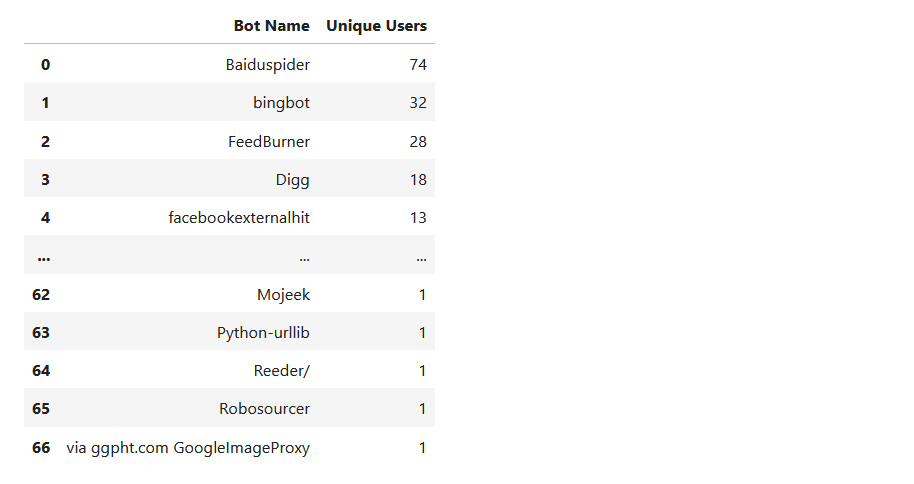
# D) Sort country by users  
top\_country = df.groupby("Country")["IP"].nunique().sort\_values(ascending = *False*)  
top\_country = top\_country.reset\_index().rename(columns = {"IP": "Unique Users"})  
top\_country.to\_csv("output/D\_top\_countries.csv", index = *False*)  
display(top\_country)



## e. Виокремити пошукових ботів

Використаємо модуль crawlerdetect, щоб дізнатися назву бота (якщо він взагалі є) за User-Agent. Потім залишимо тільки ті записи, що містять назву бота, згрупуємо за цими назвами і відсортуємо за спаданням:

# E) Select search bots  
*def* get\_bot\_name(*agent*):  
 bot\_name = ""  
 crawler\_detect = CrawlerDetect()  
 *if* crawler\_detect.isCrawler(*agent*): # Check if user agent is a bot  
 bot\_name = crawler\_detect.getMatches() # Get the name of the bot  
 *if not* bot\_name:  
 bot\_name = "Unknown"  
 *return* bot\_name  
  
search\_bots = df.copy(deep = *True*)  
search\_bots["Bot Name"] = search\_bots["User-Agent"].apply(get\_bot\_name)  
search\_bots = search\_bots[search\_bots["Bot Name"] != ""]  
search\_bots = search\_bots.groupby("Bot Name")["IP"].nunique().sort\_values(ascending = *False*)  
search\_bots = search\_bots.reset\_index().rename(columns = {"IP": "Unique Users"})  
search\_bots.to\_csv("output/E\_search\_bots.csv", index = *False*)  
display(search\_bots)



## f. Детектувати аномалії (якщо такі є)

Створимо об'єкт datetime, об'єднавши дату та час, щоб бачити таймлайн. Обчислимо Z-score для значень фічі Size (можемо додати інші чисельні фічі, що отримати оцінку і для них).

Аномалії – це точки, модуль оцінки яких більше 3, тобто |Z-score| > 3

# F) Detect anomalies  
# Get datetime objects from separated date & time  
datetime = pd.to\_datetime(df["Date"] + " " + df["Time"], format = "%Y-%m-%d %H:%M:%S")  
  
# Calculate Z-score for numerical features  
cols = ["Size"] # Status can be added  
*for* col *in* cols:  
 df\_zscore = df[[col]].apply(zscore)  
 threshold = 3 # Z-score that won"t be treated as anomaly is in range [-3; 3]  
 anomalies = df[(df\_zscore > threshold).any(axis = 1)]  
 anomalies.to\_csv(f"output/F\_{col}\_anomalies.csv", index = *False*)  
 print(f"Found {len(anomalies)} anomalies")  
  
 # Plot anomalies  
 plt.figure(figsize = (10, 5))  
 plt.plot(datetime, df[col], label = col, color = "blue", alpha = 0.25)  
 plt.scatter(datetime[anomalies.index], anomalies[col], color = "green", label = "Anomalies", marker = "\*")  
 plt.title(f"{col} with anomalies")  
 plt.xlabel("Datetime")  
 plt.ylabel(col)  
 plt.grid(*True*)  
 plt.tight\_layout()  
 plt.legend()  
 plt.show()

Знайдено 45 аномалій:

