EKONOMICKÁ UNIVERZITA V BRATISLAVE
FAKULTA HOSPODÁRSKEJ INFORMATIKY

Analýza neštrukturovaného a semištrukturovaného datasetu pomocou Sparku

Zadanie z predmetu Big Data

Bc. Ondrej Šima

Bc. Alena Stracenská

2023

Obsah

Zoznam obrázkov a tabuliek

Úvod

Spark bol vyvinutý na Univerzite Kalifornie v Berkeley ako náhrada za Hadoop MapReduce a ponúka niekoľko výhod, ako napríklad rýchlejšie spracovanie dát, podpora množstva dátových zdrojov a možnosť spracovania úloh v reálnom čase.

Pyspark je Python rozhranie pre Spark, ktoré umožňuje používanie Sparku s Pythonom. Je ho vhodné používať pre spracovanie veľkých objemov dát a paralelizáciu úloh, ktoré by boli ťažko spracovateľné na jednom stroji. Umožňuje spracovanie rôznych typov dát, ako sú štruktúrované, neštruktúrované alebo pološtruktúrované dáta.

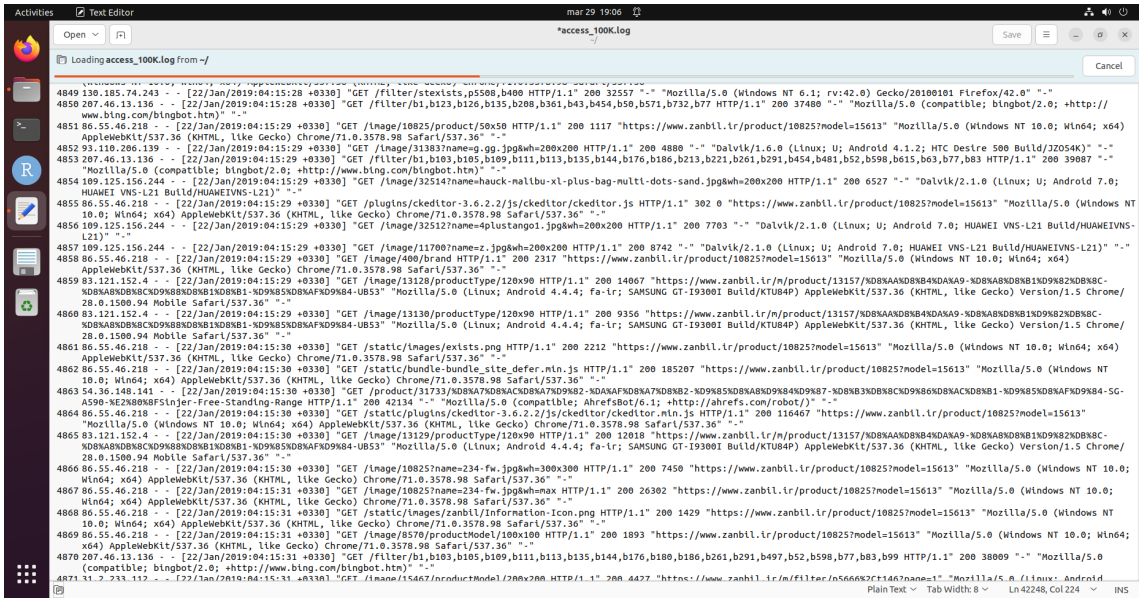
V tomto zadaní si napíšeme vlastný `sparkapp.py` skript, ktorý si prispôsobíme podľa toho, aký výstup chceme dosiahnuť. Následne si ho zapíšeme do výsledného súboru, v ktorom si správnosť spracovania logov overíme.

Veríme, že tento dokument bude prínosný pre čitateľa nie len po teoretickej stránke, ale aj po praktickej, kde sa naučí ako možno spracovať dáta pomocou Sparku resp. PySparku.

1 Analýza semištrukturovaného datasetu

1.1 Použité dáta

Dáta, s ktorými sme pracovali, nie len teraz, ale aj v predošlých zadaniach, boli vo forme apačovských webových logov. Môžeme si ich stiahnuť na kaggle.com. Sú to pološtruktúrované dáta, ktorých veľkosť obmedzíme na 100 000 riadkov.



Obrázok 1: Ukážka súboru s logmi, zdroj: [vlastné spracovanie]

1.2 Tvorba `sparkapp.py` skriptu

Najprv sme stiahli `spark-3.3.2-bin-hadoop3.tgz` pre nami použitú verziu Hadoopu. Po rozbalení a nastavení systémovej premennej PATH sme nainštalovali PySpark cez [1]:

```
$ pip install pyspark
$ pyspark
```

V zdrojovom kóde sme najskôr naimportovali potrebné knižnice, následne nastavili Spark reláciu. Súbor budeme načítavať do Spark DataFrame. Najskôr sme pre súbor vytvorili schému, čiže priradili jednotlivým poliam logu ktorý budeme parsovať, názov a typ. Načítaný log zo súboru potom parsujeme do jednotlivých polí DataFrame pomocou

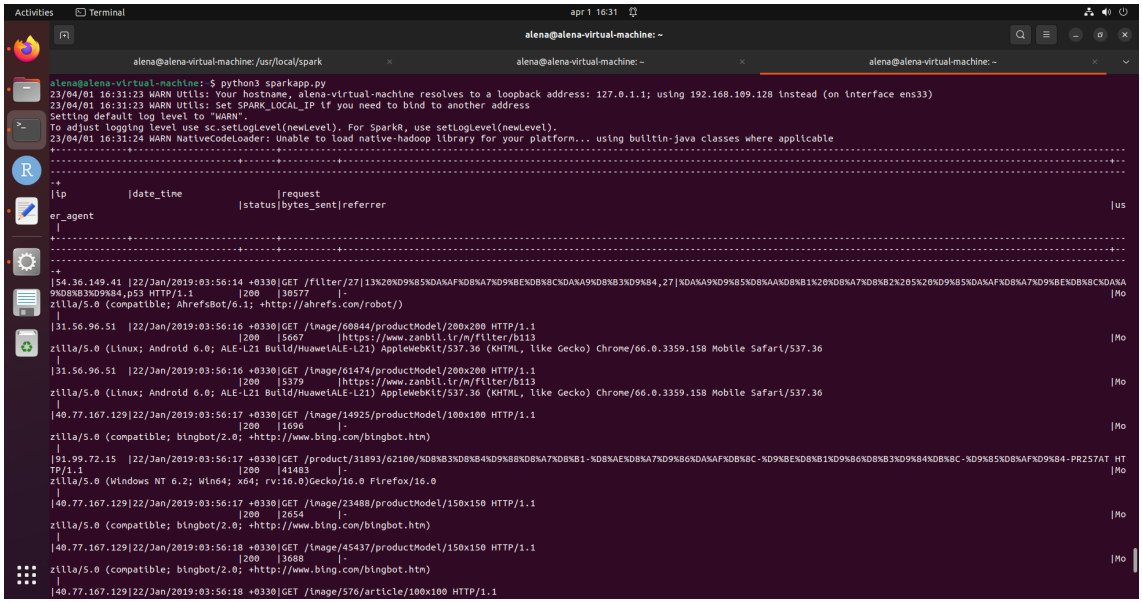
regexov. Spracovaný log sme potom uložili ako štrukturovaný `.csv` súbor a zobrazili. Kód v skripte`sparkapp.py` je nasledovný:

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import split, regexp_extract
from pyspark.sql.types import StructType, StructField, StringType, IntegerType

# Spark relacia
spark = SparkSession.builder.appName("LogParser").getOrCreate()

#cesta k suboru
log_file_path = "/home/alena/access_100K.log"

# nastav schemu
schema = StructType([
    StructField("ip", StringType(), True),
    StructField("date_time", StringType(), True), #zatial ako string
    StructField("request", StringType(), True),
    StructField("status", IntegerType(), True),
    StructField("bytes_sent", IntegerType(), True),
    StructField("referrer", StringType(), True),
    StructField("user_agent", StringType(), True),
    StructField("other", StringType(), True)])

#nacitanie zo suboru
log_df = spark.read.text(log_file_path)

#parsovanie do poli
parsed_df = log_df.select(
    regexp_extract('value', r'^([^\s]+)', 1).alias('ip'),
    regexp_extract('value', r'\[(.*)\]', 1).alias('date_time'),
    regexp_extract('value', r'"([^"]+)"', 1).alias('request'),
    regexp_extract('value', r'\s(\d{3})\s', 1).cast(IntegerType()).alias('status'),
    regexp_extract('value', r'\s(\d+) \"', 1).cast(IntegerType()).alias('bytes_sent'),
    regexp_extract('value', r'"([^"]+)"\s+"[^"]*"', 1).alias('referrer'),
    regexp_extract('value', r'"[^"]+"\s+"([^"]*)"', 1).alias('user_agent'),
    regexp_extract('value', r'\s+"[^"]+"$', 0).alias('other')
).drop("other")

parsed_df.write.csv("/home/alena/parsed_logs.csv")
parsed_df.show(truncate=False)
```

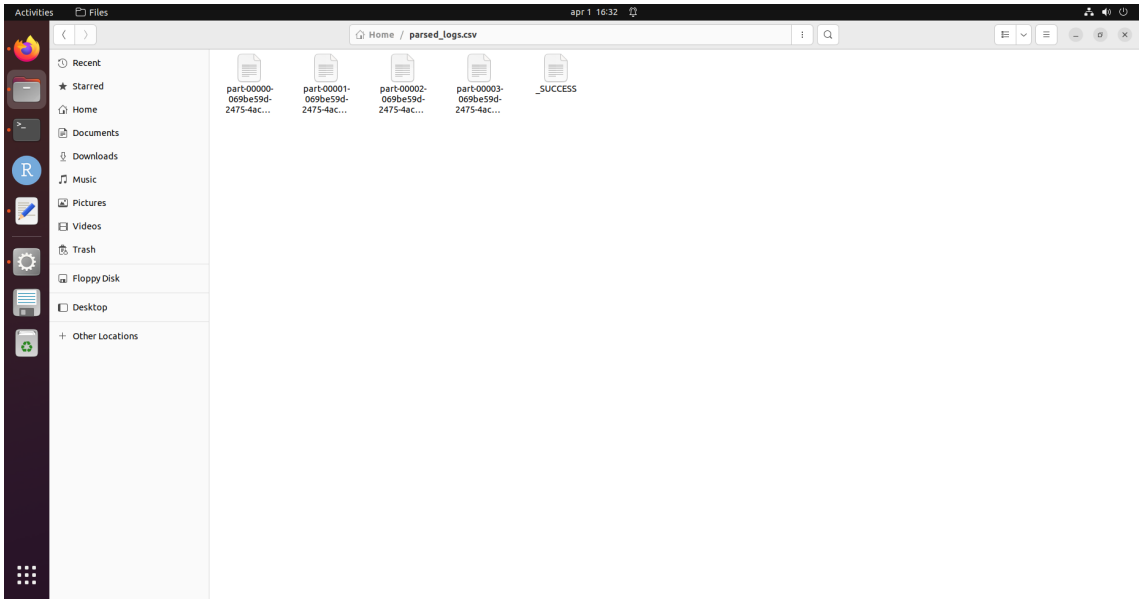
Následne sme spustili daný skript pomocou príkazu:

```
python3 sparkapp.py
```

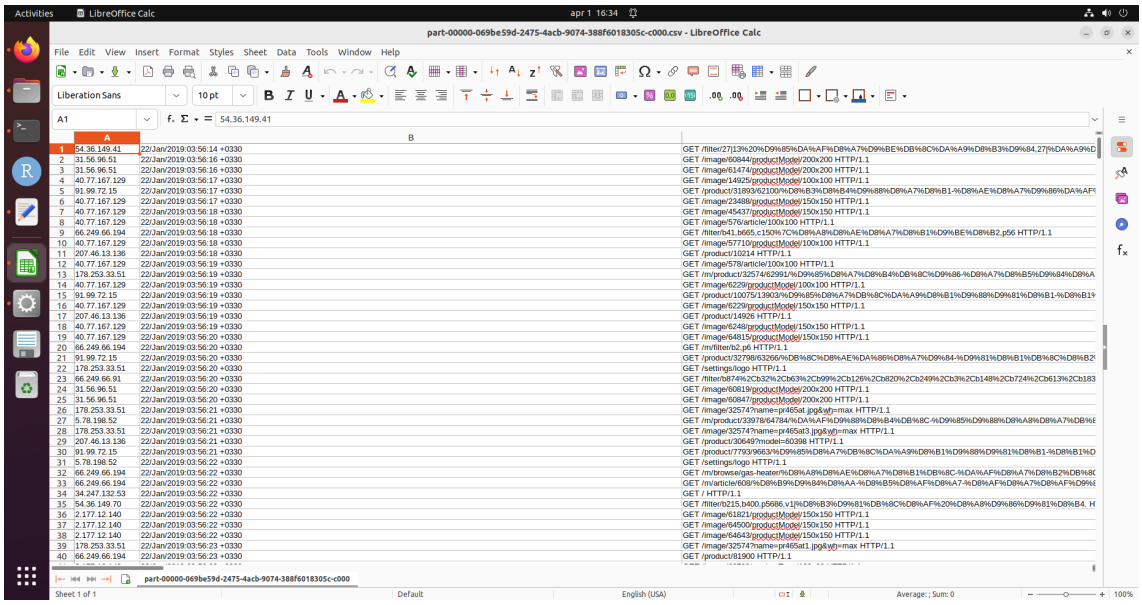
Výstupom programu je náhľad (prvých 20 riadkov) z vyparsovaného súboru a vyparsovaný, štrukturovaný `.csv` súbor.



Obrázok 2: Spustenie `sparkapp.py` skriptu, zdroj: [vlastné spracovanie]



Obrázok 3: Ukážka výsledných .csv súborov, zdroj: [vlastné spracovanie]



Obrázok 4: Obsah `.csv` súboru, zdroj: [vlastné spracovanie]

Záver

Cieľom tohto zadania bolo opísať tvorbu `sparkapp.py` skriptu a logiku jeho vykonávania. Následne ho spustiť a overiť, či výstup, ktorý sme chceli bol dosiahnutý. Cieľ tohto zadania sme splnili.

Problémy sa nám vyskytli najmä pri tvorbe skriptu, kde nám trvalo dlhší čas pochopiť, ako sa pracuje so Spark DataFrame, ktorý sa líši od klasického Pandas DataFrame. Môžeme tak povedať, že logy sme spracovali a rozparsovali správne podľa jednotlivých kategórií a týmto deklarovali funkčnosť uvedeného Python kódu v skripte.

Veríme, že toto zadanie prinesie čitateľovi nový pohľad na možnosť spracovávať veľké súbory a získa nové prípadne obohatí už doterajšie vedomosti a schopnosti o tvorbu takýchto skriptov.

Zoznam použitej literatúry

[1] Apache Software Foundation. 2023. *Download Apache Spark™* In spark.apache.org [online]. 2023. [citované dňa 1.4.2023]. Dostupné na internete: `https://spark.apache.org/downloads.html`.