**Assignment – 2**

**CWID:** 50380788  
**Name:** Mukesh Ravichandran

## Source Code and Output Screenshots

### **Environment Setup**

# Set the PySpark environment variables

from pyspark import SparkContext, SparkConf

from operator import add

import re

import os

os.environ["PYSPARK\_PYTHON"] = r"C:\Users\mukeshravichandran\AppData\Local\Programs\Python\Python310\python.exe"

import datetime

from pyspark.sql import Row

import matplotlib.pyplot as plt

# Initialize SparkContext

sc = SparkContext("local", "NASA RDD")

### **Log Parsing Logic**

def parseApacheLogLine(logline):

    """ Parse a line in the Apache Common Log format

    Args:

        logline (str): a line of text in the Apache Common Log format

    Returns:

        tuple: either a dictionary containing the parts of the Apache Access Log and 1,

               or the original invalid log line and 0

    """

    match = re.search(APACHE\_ACCESS\_LOG\_PATTERN, logline)

    if match is None:

        return (logline, 0)

    size\_field = match.group(9)

    if size\_field == '-':

        size = int(0)

    else:

        size = int(match.group(9))

    return (Row(

        host          = match.group(1),

        client\_identd = match.group(2),

        user\_id       = match.group(3),

        date\_time     = parse\_apache\_time(match.group(4)),

        method        = match.group(5),

        endpoint      = match.group(6),

        protocol      = match.group(7),

        response\_code = int(match.group(8)),

        content\_size  = size

    ), 1)

 # A regular expression pattern to extract fields from the log line

APACHE\_ACCESS\_LOG\_PATTERN = '^(\S+) (\S+) (\S+) \[([^\]]+)\] "(\S+) (\S+) (\S+)" (\d{3}) (\S+)$'

Configuration and RDD creation

A screenshot of a computer program

AI-generated content may be incorrect.

### **Initial Data Cleaning**

630 records failed to parse. These were addressed by refining the regular expression to ensure consistent parsing.

A computer screen with text

AI-generated content may be incorrect.

### **Statistics and Response Code Analysis**

Content size statistics and response code frequencies were computed using reduceByKey with lambda expressions.

A screen shot of a computer program

AI-generated content may be incorrect.

### **Response Code Visualization**

#  (2c) Example: Response Code Graphing with matplotlib

labels = responseCodeToCount.map(lambda pair: pair[0]).collect()

print(labels)

count = access\_logs.count()

fracs = responseCodeToCount.map(lambda pair: float(pair[1]) / count).collect()

print(fracs)

def pie\_pct\_format(value):

    """ Determine the appropriate format string for the pie chart percentage label

    Args:

        value: value of the pie slice

    Returns:

        str: formated string label; if the slice is too small to fit, returns an empty string for label

    """

    return '' if value < 7 else '%.0f%%' % value

fig = plt.figure(figsize=(4.5, 4.5), facecolor='white', edgecolor='white')

colors = ['yellowgreen', 'lightskyblue', 'gold', 'purple', 'lightcoral', 'yellow', 'black']

explode = (0.05, 0.05, 0.1, 0, 0, 0, 0)

patches, texts, autotexts = plt.pie(fracs, labels=labels, colors=colors,

                                    explode=explode, autopct=pie\_pct\_format,

                                    shadow=False,  startangle=125)

for text, autotext in zip(texts, autotexts):

    if autotext.get\_text() == '':

        text.set\_text('')  # If the slice is small to fit, don't show a text label

plt.legend(labels, loc=(0.80, -0.1), shadow=True)

A screenshot of a computer

AI-generated content may be incorrect.

Analyzing the frequent hosts that have accessed the server

A computer screen shot of a black screen

AI-generated content may be incorrect.

A black screen with colorful text

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

### **Endpoint Analysis and Visualization**

Utilized reduceByKey to rank endpoints. Top 10 were visualized using a matplotlib plot.

A screenshot of a computer

AI-generated content may be incorrect.

### **Error Endpoint Detection**

Filtered records where response code ≠ 200 and sorted them by frequency to isolate the top 10 problematic endpoints.

A screenshot of a computer

AI-generated content may be incorrect.

### **Unique Host Count**

Calculated the number of distinct hosts across the logs.

A screenshot of a computer program

AI-generated content may be incorrect.

### **Daily Unique Host Visualization**

# (3d) Exercise: Visualizing the Number of Unique Daily Hosts

daysWithHosts = dailyHosts.map(lambda x: x[0]).collect()

hosts = dailyHosts.map(lambda x: x[1]).collect()

print(daysWithHosts)  # [2, 3, 4, 5, ...]

print(hosts)          # [4859, 7336, 5524, 7383, ...]

fig = plt.figure(figsize=(8, 4.5), facecolor='white', edgecolor='white')

plt.axis([min(daysWithHosts), max(daysWithHosts), 0, max(hosts) + 500])

plt.grid(visible=True, which='major', axis='y')

plt.xlabel('Day')

plt.ylabel('Hosts')

plt.plot(daysWithHosts, hosts, marker='x', color='red')

plt.title('Unique Daily Hosts in July 1995')

plt.show()

A graph with red line

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

### **404 Response Analysis**

Explored all records that returned a 404 response.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a graph

AI-generated content may be incorrect.

A computer screen shot of a code

AI-generated content may be incorrect.

### **Final Steps**

Visualized the findings and gracefully terminated the Spark context.

A screen shot of a graph

AI-generated content may be incorrect.

## **Challenges and Resolutions**

1. **PySpark Module Not Found in Jupyter**
   * **Resolution:** Installed PySpark in Jupyter, but due to further issues, switched to VS Code where the environment was already configured correctly.
2. **AttributeError for Fields like log.ip or log.date\_time**
   * **Resolution:** This error occurred because the raw text was not parsed into structured Row objects. Solved by applying the parseLogs() function before transformations.
3. **SparkContext Initialization Fails (ConnectionRefusedError)**
   * **Resolution:** Resolved by checking for existing SparkContexts and stopping them before starting a new one.
4. **Grid Plotting Error in Matplotlib**
   * **Issue:** Using plt.grid(b=True) raised a warning.
   * **Fix:** Replaced with visible=True as b is deprecated.