Name:Raj Kariya
ID:- 202103048

Colab Link:-
https://colab.research.google.com/drive/1EfBlbOiV4IZylxQCBbT5WYzhYn3zlERa?usp=sharing
Q1)Find the most popular browser (made most requests) from the “web access log” file of lab01. You can know about the position of the agent in weblog from https://httpd.apache.org/docs/2.4/logs.html#accesslog

Code:-

```
%%file PopWebAccess.py
from mrjob.job import MRJob
from mrjob.job import MRStep
import re

class PopWebSummary(MRJob):
    # key is browserName and value is 1
    def mapper(self, _ , line):
      record = line.split(' ')
      browserName = record[-2]
      yield browserName,1

    #reducer function takes input as key-value pairs and return frquency
with browserName
    def reducer(self, browserName, count):
      yield None,(sum(count),browserName)

    #Gives the most frequent browser
    def GiveTheMostPopular(self,_,valuePair):
      yield max(valuePair)

    def steps(self):
      return [

MRStep(mapper=self.mapper,reducer=self.reducer),MRStep(reducer=self.GiveTh
eMostPopular)
      ]
if __name__ == '__main__':
    PopWebSummary.run()
```

Output:

```
[26] !python PopWebAccess.py "/content/gdrive/MyDrive/Colab Notebooks/datasets/mr/web_access_log.txt"

No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/PopWebAccess.root.20230823.154059.566509
Running step 1 of 2...
Running step 2 of 2...
job output is in /tmp/PopWebAccess.root.20230823.154059.566509/output
Streaming final output from /tmp/PopWebAccess.root.20230823.154059.566509/output...
60234   "Safari/537.36\""
Removing temp directory /tmp/PopWebAccess.root.20230823.154059.566509...
```

Q2)Perform the following task, a simple classification approach. Suppose you are given a set of data vectors in the file iris.csv. Read each row in data file as a data vector. You are also given a file containing class vectors in iris_classes. Read each row in this file as a class vector. Both data files are available under iris subfolder of the dataset folder shared with you. [save this link for future reference]
https://drive.google.com/drive/folders/1Q0sy0NlD2nkjmzxuYURQoFt5XRZpcScs Your computation task goes as following: Iterate through all data vector and determine nearest class vector for each data vector. Let nearness be computed by euclidean distance between data vector and class vector. Your program should output ID of class vector for each data vector. Let ID of class vector be 1,2,3 in the order of their occurrence in the class file. Hint: It can be implemented as map only task. Load class vectors in “INIT” method of mapper. Store them in class level field that can be used in mapper method.

Code:-

```
%%file NearestDistance.py
import math
from mrjob.job import MRJob
import re
import csv
def is_float(num):
  try:
    float(num)
    return True
  except ValueError:
    return False
class NearestDistance(MRJob):

    # mapper init function which loads the class vectors
    def mapper_init(self):
      self.classes = { }
```

```
      iter = 0
      with open('/content/gdrive/MyDrive/Colab
Notebooks/datasets/iris/iris.csv',mode = 'r') as file:
        for data in csv.reader(file):
          if (is_float(data[0])):
            iter += 1
            data = ([float(k) for k in data])
            self.classes[iter] = data

    #
    def mapper(self,key,line):
      record = re.split(',',line)
      minDistance = math.inf
      ID = 0
      if (is_float(record[0])):
        record = [float(k) for k in record]
        for classID,classVector in self.classes.items():
          d = 0
          for i in range(len(classVector)):
            d += (classVector[i] - record[i])**2
          if (minDistance > math.sqrt(d)):
            minDistance = math.sqrt(d)
            ID = classID
        yield record,ID
if __name__ == '__main__':
    NearestDistance.run()
```

Output:

```
[68] !python NearestDistance.py "/content/gdrive/MyDrive/Colab Notebooks/datasets/iris/iris.csv"
```

```
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/NearestDistance.root.20230823.165450.080978
Running step 1 of 1...
job output is in /tmp/NearestDistance.root.20230823.165450.080978/output
Streaming final output from /tmp/NearestDistance.root.20230823.165450.080978/output...
[6.4, 2.9, 4.3, 1.3]    75
[6.6, 3.0, 4.4, 1.4]    76
[6.8, 2.8, 4.8, 1.4]    77
[6.7, 3.0, 5.0, 1.7]    78
[6.0, 2.9, 4.5, 1.5]    79
[5.7, 2.6, 3.5, 1.0]    80
[5.5, 2.4, 3.8, 1.1]    81
[5.5, 2.4, 3.7, 1.0]    82
[5.8, 2.7, 3.9, 1.2]    83
[6.0, 2.7, 5.1, 1.6]    84
[5.4, 3.0, 4.5, 1.5]    85
[6.0, 3.4, 4.5, 1.6]    86
[6.7, 3.1, 4.7, 1.5]    87
[6.3, 2.3, 4.4, 1.3]    88
[5.6, 3.0, 4.1, 1.3]    89
[5.5, 2.5, 4.0, 1.3]    90
[5.5, 2.6, 4.4, 1.2]    91
[6.1, 3.0, 4.6, 1.4]    92
[5.8, 2.6, 4.0, 1.2]    93
[5.0, 2.3, 3.3, 1.0]    94
[5.6, 2.7, 4.2, 1.3]    95
[5.7, 3.0, 4.2, 1.2]    96
[5.7, 2.9, 4.2, 1.3]    97
[6.2, 2.9, 4.3, 1.3]    98
[5.1, 2.5, 3.0, 1.1]    99
[5.7, 2.8, 4.1, 1.3]    100
[6.3, 3.3, 6.0, 2.5]    101
[5.8, 2.7, 5.1, 1.9]    102
[7.1, 3.0, 5.9, 2.1]    103
[6.3, 2.9, 5.6, 1.8]    104
```

Q3)Compute top 10 earners from the “employee.csv” of lab01. You can only list employee numbers. For computing top-10, you can maintain “sorted map” with size of 10 at mappers. Can refer to “lecture slides”.

Code:

```
%%file Earners.py
import math
from mrjob.job import MRJob
import re
import heapq as h
class Earners(MRJob):
```

```
    def mapper_init(self):
      self.N = 10
      self.minHeap = []
      h.heapify(self.minHeap)
    def mapper(self,key,line):
      record = re.split(',',line)
      if(record[0] != '\ufeff10026'):
        salary = int(record[3])
        Emp_no = int(record[2])
        pair = (salary,Emp_no)
        h.heappush(self.minHeap,pair)
      if (len(self.minHeap) > self.N):
        h.heappop(self.minHeap)
    def mapper_final(self):
      for i in list(self.minHeap):
        yield None,i
    def reducer_init(self):
      self.N = 10
      self.minHeap = []
      h.heapify(self.minHeap)
    def reducer(self,_,value_pairs):
      for i in value_pairs:
        h.heappush(self.minHeap,i)
        if(len(self.minHeap) > self.N):
          h.heappop(self.minHeap)
    def reducer_final(self):
      for i in range(self.N):
        yield i+1,list(self.minHeap)[self.N - i - 1][1]
if __name__ == '__main__':
    Earners.run()
```

Output:

```
!python Earners.py "/content/gdrive/MyDrive/Colab Notebooks/datasets/mr/employee.csv"
```

```
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/Earners.root.20230823.175609.001992
Running step 1 of 1...
job output is in /tmp/Earners.root.20230823.175609.001992/output
Streaming final output from /tmp/Earners.root.20230823.175609.001992/output...
1	2
2	6
3	3
4	3
5	3
6	3
7	5
8	3
9	3
10	3
Removing temp directory /tmp/Earners.root.20230823.175609.001992...
```

Q4)Compute the standard deviation of salary for each department from “employee.csv”

Code:

```
%%file StdDev.py
import math
from mrjob.job import MRJob
import re
import heapq as h
import statistics
class StdDev(MRJob):
    def mapper(self,key,line):
      record = re.split(',',line)
      dno = record[2]
      salary = int(record[3])
      yield dno,salary
    def reducer(self,dno,salaries):
      List = list(salaries)
      n = len(List)
      StdDeviation = 0
      mean = sum(List)/n
      for i in List:
        StdDeviation += (i - mean)**2
      yield dno,math.sqrt(StdDeviation/n)
if __name__ == '__main__':
```

```
    StdDev.run()
```

Output:

```
[84] !python StdDev.py "/content/gdrive/MyDrive/Colab Notebooks/datasets/mr/employee.csv"

No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/StdDev.root.20230823.180924.053792
Running step 1 of 1...
job output is in /tmp/StdDev.root.20230823.180924.053792/output
Streaming final output from /tmp/StdDev.root.20230823.180924.053792/output...
"6"     20702.734864531638
"1"     19722.68484258672
"2"     0.0
"3"     32875.83877242373
"4"     8754.93245490792
"5"     11420.800779947584
Removing temp directory /tmp/StdDev.root.20230823.180924.053792...
```