732A54: Big Data Analytics - Lab BDA1 – Spark

Umamaheswarababu Maddela (umama339)
Dinesh Sundaramoorthy (dinsu875)

Assignment-1:

What are the lowest and highest temperatures measured each year for the period 1950-2014. Provide the lists sorted in the descending order with respect to the maximum temperature. In this exercise you will use the temperature-readings.csv file. The output should at least contain the following information (You can also include a Station column so that you may find multiple stations that record the highest (lowest) temperature.):
Year, temperature

Solution:

For Highest Temperatures:

Code:

```
# To print maximum temperatures

from pyspark import SparkContext

def max_temperature(a,b):
    if a>=b:
        return a
    else:
        return b

sc = SparkContext(appName = "assignment_1")
temperature_file= sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temperature = lines.map(lambda x: (x[1][0:4], float(x[3])))
year_temperature = year_temperature.filter(lambda x: int(x[0])>=1950 and int(x[0])<=2014)

max_temperatures =year_temperature.reduceByKey(max_temperature)
temperatures = max_temperatures.sortBy(ascending = False, keyfunc = lambda x : x[1])
temperatures.saveAsTextFile("BDA/output")
```

Output: (First 5 lines)

(u'1975', 36.1)
(u'1992', 35.4)
(u'1994', 34.7)
(u'2014', 34.4)
(u'2010', 34.4)

For Lowest Temperatures:

Code:

```
# To print minimum temperatures

from pyspark import SparkContext

def min_temperature(a,b):
    if b>=a:
        return a
    else:
        return b

sc = SparkContext(appName = "assignment_1")
temperature_file= sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temperature = lines.map(lambda x: (x[1][0:4], float(x[3])))
year_temperature = year_temperature.filter(lambda x: int(x[0])>=1950 and int(x[0])<=2014)

min_temperatures =year_temperature.reduceByKey(min_temperature)
temperatures = min_temperatures.sortBy(ascending = True, keyfunc = lambda x : x[1])
temperatures.saveAsTextFile("BDA/output")
```

Output: (First 5 lines)

(u'1966', -49.4)
(u'1999', -49.0)
(u'1978', -47.7)
(u'1987', -47.3)
(u'1967', -45.4)

Sorted in the descending order with respect to the maximum temperature:

Code:

```
from pyspark import SparkContext

def max_temperature(a,b):
    if a>=b:
        return a
    else:
        return b

def min_temperature(a,b):
    if b>=a:
        return a
    else:
        return b

sc = SparkContext(appName = "assignment_1")
temperature_file= sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
year_temperature = lines.map(lambda x: (x[1][0:4], float(x[3])))
year_temperature = year_temperature.filter(lambda x: int(x[0])>=1950 and int(x[0])<=2014)

max_temperatures =year_temperature.reduceByKey(max_temperature)
min_temperatures =year_temperature.reduceByKey(min_temperature)

temperatures = max_temperatures.join(min_temperatures)
temperatures = temperatures.sortBy(ascending = False, keyfunc = lambda x : x[1][0])
temperatures.saveAsTextFile("BDA/output")
```

Output: (First 20 lines)

(u'1975', (36.1, -37.0))
(u'1992', (35.4, -36.1))
(u'1994', (34.7, -40.5))
(u'2014', (34.4, -42.5))
(u'2010', (34.4, -41.7))
(u'1989', (33.9, -38.2))
(u'1982', (33.8, -42.2))
(u'1968', (33.7, -42.0))
(u'1966', (33.5, -49.4))
(u'2002', (33.3, -42.2))
(u'1983', (33.3, -38.2))
(u'1986', (33.2, -44.2))
(u'1970', (33.2, -39.6))
(u'1956', (33.0, -45.0))
(u'2000', (33.0, -37.6))
(u'1959', (32.8, -43.6))
(u'2006', (32.7, -40.6))
(u'1991', (32.7, -39.3))
(u'1988', (32.6, -39.9))
(u'2011', (32.5, -42.0))

Assignment-2:

Count the number of readings for each month in the period of 1950-2014 which are higher than 10 degrees. Repeat the exercise, this time taking only distinct readings from each station. That is, if a station reported a reading above 10 degrees in some month, then it appears only once in the count for that month.

In this exercise you will use the temperature-readings.csv file. The output should contain the following information:

Year, month, count

Solution:

Number of readings for each month in the period of 1950-2014 which are higher than 10 degrees:

Code:

```
from pyspark import SparkContext

sc = SparkContext(appName="assignment_2")
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

year_temperature = lines.map(lambda x: (x[1][0:7], float(x[3])))

year_temperature = year_temperature.filter(lambda x: int(x[0][0:4]) >= 1950 and int(x[0][0:4]) <= 2014)
year_temperature = year_temperature.filter(lambda x: (x[1] > 10))

count = year_temperature.map(lambda x: (x[0], 1))
count = count.reduceByKey(lambda x,y: x+y)

count_sort = count.sortBy(ascending = False, keyfunc=lambda x: x[1])

count_sort.saveAsTextFile("BDA/output")
```

Output: (First 20 lines)

(u'2014-07', 147681)
(u'2011-07', 146656)
(u'2010-07', 143419)
(u'2012-07', 137477)
(u'2013-07', 133657)
(u'2009-07', 133008)
(u'2011-08', 132734)
(u'2009-08', 128349)
(u'2013-08', 128235)
(u'2003-07', 128133)
(u'2002-07', 127956)
(u'2006-08', 127622)
(u'2008-07', 126973)
(u'2002-08', 126073)
(u'2005-07', 125294)
(u'2011-06', 125193)
(u'2012-08', 125037)
(u'2006-07', 124794)
(u'2010-08', 124417)
(u'2014-08', 124045)

Taking only distinct readings from each station:

Code:

```
sc = SparkContext(appName = "assignment_2a")

temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

year_temperature = lines.map(lambda x:((x[0],x[1][0:7],1), float(x[3])))

filtered_temperature = year_temperature.filter(lambda x: int(x[0][1][0:4]) >= 1950 and int(x[0][1][0:4]) <= 2014)

filtered_temperature = filtered_temperature.filter(lambda x: x[1]>10)
distinct_temperature = filtered_temperature.map(lambda x:(x[0])).distinct()
distinct_temperature = distinct_temperature.map(lambda x:(x[1],x[2]))
count= distinct_temperature.reduceByKey(lambda x,y: x+y)

count_sort = count.sortBy(ascending = False, keyfunc=lambda x: x[1])
count_sort.saveAsTextFile("BDA/output")
```

Output: (First 20 lines)

(u'1972-10', 378)
(u'1973-06', 377)
(u'1973-05', 377)
(u'1972-08', 376)
(u'1973-09', 376)
(u'1972-05', 375)
(u'1971-08', 375)
(u'1972-09', 375)
(u'1972-06', 375)
(u'1971-09', 374)
(u'1972-07', 374)
(u'1971-06', 374)
(u'1971-05', 373)
(u'1973-08', 373)
(u'1974-06', 372)
(u'1974-08', 372)
(u'1973-07', 370)
(u'1970-08', 370)
(u'1974-05', 370)
(u'1971-07', 370)

Assignment-3:

Find the average monthly temperature for each available station in Sweden. Your result should include average temperature for each station for each month in the period of 1960- 2014. Bear in mind that not every station has the readings for each month in this timeframe. In this exercise you will use the temperature-readings.csv file. The output should contain the following information:

Year, month, station number, average monthly temperature

Solution:

Code:

```
from pyspark import SparkContext

sc = SparkContext(appName="assignment_3")

temperature_files = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_files.map(lambda line: line.split(";"))

year_temperature = lines.map(lambda x: ((x[1][0:7], x[0]), (float(x[3]), float(x[3]))))
year_temperature = year_temperature.filter(lambda x: int(x[0][0][0:4])>=1960 and int(x[0][0][0:4])<=2014)
year_temperature = year_temperature.reduceByKey(lambda x,y: (x[0] if x[0]<y[0] else y[0], x[1] if x[1]>=y[1] else y[1]))

count = year_temperature.map(lambda x: ((x[0][0], x[0][1]), (x[1][0] + x[1][1], 1)))
add_temp = count.reduceByKey(lambda x,y: (x[0]+y[0], x[1]+y[1]))
average_temp = add_temp.map(lambda x: (x[0][0], x[0][1], x[1][0]/ (2*x[1][1])))

output = average_temp.sortBy(ascending=False, keyfunc=lambda k: k[2])
output.saveAsTextFile("BDA/output")
```

Output: (First 20 lines)

(u'2014-07', u'96000', 26.3)
(u'1975-08', u'53560', 24.85)
(u'1975-08', u'78140', 24.75)
(u'1994-07', u'76000', 24.5)
(u'1975-08', u'86200', 24.450000000000003)
(u'1994-07', u'85280', 24.25)
(u'1975-08', u'87540', 24.099999999999998)
(u'1994-07', u'75120', 24.0)
(u'1975-08', u'98210', 23.95)
(u'1994-07', u'86330', 23.85)
(u'2002-08', u'78140', 23.8)
(u'2006-07', u'76530', 23.75)
(u'1975-08', u'97120', 23.7)
(u'1994-07', u'96000', 23.7)
(u'1975-08', u'97190', 23.7)
(u'1975-08', u'97210', 23.7)
(u'1975-08', u'85220', 23.6)
(u'1994-07', u'85210', 23.6)
(u'1994-07', u'62400', 23.6)
(u'1975-08', u'54550', 23.5)

Assignment-4:

Provide a list of stations with their associated maximum measured temperatures and maximum measured daily precipitation. Show only those stations where the maximum temperature is between 25 and 30 degrees and maximum daily precipitation is between 100 mm and 200mm. In this exercise you will use the temperature-readings.csv and precipitation-readings.csv files. The output should contain the following information:

Station number, maximum measured temperature, maximum daily precipitation

Solution:

Code:

```
from pyspark import SparkContext

sc = SparkContext(appName="assignment_4")

temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
precip_file = sc.textFile("BDA/input/precipitation-readings.csv")

temperature_lines = temperature_file.map(lambda line: line.split(";"))
stat_temperature = temperature_lines.map(lambda x: (x[0], float(x[3])))
max_temp = stat_temperature.reduceByKey(max)
filter_temp = max_temp.filter(lambda x: 25 <= x[1] <= 30)

precip_lines = precip_file.map(lambda line: line.split(";"))
stat_precipitation = precip_lines.map(lambda x: ((x[0],x[1]), float(x[3])))

#summing up daily precipitation for each station
stat_precipitation = stat_precipitation.reduceByKey(lambda x,y: x+y)

stat_precipitation = stat_precipitation.map(lambda x: (x[0][0], float(x[1])))
max_rain = stat_precipitation.reduceByKey(max)
filter_rain = max_rain.filter(lambda x: 100 <= x[1] <= 200)

output = filter_rain.join(filter_temp).map(lambda x: (x[0], x[1][1], x[1][0]))
output.saveAsTextFile("BDA/output")
```

Output:

There is no output for the given filter conditions

Assignment-5:

Calculate the average monthly precipitation for the Östergotland region (list of stations is provided in the separate file) for the period 1993-2016. In order to do this, you will first need to calculate the total monthly precipitation for each station before calculating the monthly average (by averaging over stations). In this exercise you will use the precipitation-readings.csv and stations-Ostergotland.csv files. The output should contain the following information:

Year, month, average monthly precipitation

Solution:

Code:

```
from pyspark import SparkContext

sc = SparkContext(appName = "assignment5")

station_file = sc.textFile("BDA/input/stations-Ostergotland.csv")
precip_file = sc.textFile("BDA/input/precipitation-readings.csv")

lines_precip = precip_file.map(lambda line: line.split(";"))
lines_osterg = station_file.map(lambda line: line.split(";"))

precip_rain = lines_precip.map(lambda x: ((x[0], x[1][0:7]), float(x[3])))
station_list = lines_osterg.map(lambda x: x[0]).collect()
precip_rain= precip_rain.filter(lambda x: x[0][0] in station_list)
precip_rain = precip_rain.filter(lambda x: int(x[0][1][0:4])>=1993 and int(x[0][1][0:4])<=2016)
precip_rain = precip_rain.reduceByKey(lambda x, y: x + y)
precip_rain = precip_rain.map(lambda x: (x[0][1][0:7], (x[1], 1)))

count_precip = precip_rain.reduceByKey(lambda x, y: (x[0] + y[0], x[1] + y[1]))
avg_precip = count_precip.mapValues(lambda x: (x[0] / x[1], 0))
avg_precip = avg_precip.map(lambda x: (x[0], x[1][0]))

avg_precip.saveAsTextFile("BDA/output")
```

Output: (First 20 lines)

(u'1996-11', 67.11666666666665)
(u'2008-03', 42.200000000000024)
(u'2008-10', 59.566666666666684)
(u'2014-05', 58.000000000000014)
(u'2001-11', 26.38333333333334)
(u'2011-05', 37.85)
(u'2010-09', 43.08333333333335)
(u'2010-02', 52.75000000000005)
(u'2013-08', 54.075)
(u'2002-06', 98.7833333333333)
(u'2013-05', 47.92500000000001)
(u'1998-11', 28.96666666666668)
(u'2002-03', 26.93333333333334)
(u'2013-02', 25.525000000000013)
(u'2007-08', 54.166666666666664)
(u'1995-12', 5.116666666666666)
(u'2000-06', 62.01666666666667)
(u'2007-02', 33.066666666666684)
(u'1993-11', 42.80000000000003)
(u'2009-07', 113.16666666666663)