Homework 2

Technology : Python: 2.7 Scala: 2.11 Spark: 2.2.1

Directory:

Surbhi_Batra_hw2

- Surbhi_Batra
  - Surbhi_Batra_Description
- OutputFiles
  - Surbhi_Batra_SON_Small2_case1-3
  - Surbhi_Batra_SON_Small2_case2-5
  - Surbhi_Batra_SON_MovieLens.Small_case1-120.txt
  - Surbhi_Batra_SON_MovieLens.Small_case1-150.txt
  - Surbhi_Batra_SON_MovieLens.Small_case2-180.txt
  - Surbhi_Batra_SON_MovieLens.Small_case2-200.txt
  - Surbhi_Batra_SON_MovieLens.Big_case1-30000.txt
  - Surbhi_Batra_SON_MovieLens.Big_case1-35000.txt
  - Surbhi_Batra_SON_MovieLens.Big_case2-2800.txt
  - Surbhi_Batra_SON_MovieLens.Big_case2-3000.txt
- Solution
  - Surbhi_Batra_SON.py

Algorithm:

Implemented the SON Algorithm.
Map Task 1 : uses the Apriori algorithm to generate the Candidate itemsets which are frequent in the chunk in the format (candidateTuple, 1)
Reduce Task 1 : collects the candidate items in a list.
Map Task 2 : counts the candidates in the chunks.
Reduce Task 2 : counts the candidate pairs and returns the frequent itemsets.

>>>Execute these Commands after : cd/to/Surbhi_Batra_hw2/Solution

Python:
Command :
$SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py <CASENUMBER> <INPUTFILE.csv> <SUPPORT>

PROBLEM 1

For **best results** in problem 1, the program works best and runs in the least possible time if no of partitions are 1. It is because the dataset is very small.

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 1 Small2.csv 3
    - Total time 36.9639399052 seconds
    - Total frequent tuples : 11300

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 2 Small2.csv 5
    - Total time 11.4254529476 seconds
    - Total frequent tuples : 5446

PROBLEM 2:

CASE 1		CASE 2	
SUPPORT THRESHOLD	**EXECUTION TIME (seconds)**	**SUPPORT THRESHOLD**	**EXECUTION TIME (seconds)**
120	6.08	180	38.06
150	5.52	200	22.75

- $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 1 MovieLens.Small.csv 120
- Total time 6.08 seconds when partitions =2 i.e. local[2] or local[*]
- Total time 6.18635201454 seconds when partitions =1 local[1]
- Total frequent tuples : 591

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 1 MovieLens.Small.csv 150
- Total time 5.52492785454 seconds when partitions =2 i.e. local[2] or local[*]
- Total frequent tuples : 144

- $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 2 MovieLens.Small.csv 180
- Total time 38.06 seconds when partitions =2 i.e. local[2] or local[*]
- Total frequent tuples : 3453

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 2 MovieLens.Small.csv 200
- Total time : 22.7473111153 seconds when partitions =2 i.e. local[2] or local[*]
- Total frequent tuples : 2137

PROBLEM 3

CASE 1		CASE 2	
SUPPORT THRESHOLD	EXECUTION TIME (seconds)	SUPPORT THRESHOLD	EXECUTION TIME (seconds)
30000	275.28	2800	347.68
35000	261.42	3000	345.30

- Command: $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 1 MovieLens.Big.csv 30000
- Total time 275.285174847 seconds when partitions =16 i.e. local[2] or local[*]
- Total frequent tuples : 207

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 1 MovieLens.Big.csv 35000
- Total time 261.426958084 seconds when partitions =16 i.e. local[2] or local[*]
- Total frequent tuples : 79

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 2 MovieLens.Big.csv 2800
- Total time 347.685868979 seconds when partitions = 16 i.e. local[2] or local[*]
- Total frequent tuples : 153

- Command : $SPARK_HOME/bin/spark-submit Surbhi_Batra_SON.py 2 MovieLens.Big.csv 3000
- Total time : 345.301845074 seconds when partitions = 16 i.e. local[2] or local[*]
- Total frequent tuples : 109