Steps to Follow:

Spark 2.3.1
Python 2.7

TASK1

Task 1 is done in python. spark-submit “/Path/to/Folder/Krish_Mehta_SONApriori” “/InputFile” “support” “/Output/path”
E.g. spark-submit Krish_Mehta_SON.py "/Users/krishmehta/Desktop/DataMining/Krish_Mehta_HW3/hw3/Data/yelp_reviews_test.txt" "40" " Krish_Mehta_SON_yelp_reviews_test_40.txt"

TASK2

Task 2 is done in python. spark-submit “/Path/to/Folder/Krish_Mehta_SONApriori” “/InputFile_small” “support” “/Output/path”
E.g. spark-submit Krish_Mehta_SON.py "/Users/krishmehta/Desktop/DataMining/Krish_Mehta_HW3/hw3/Data/yelp_reviews_small.txt" "500" " Krish_Mehta_SON_yelp_reviews_small_500.txt"

TASK3

Task 3 is done in python. spark-submit “/Path/to/Folder/Krish_Mehta_SONApriori” “/InputFile_large” “support” “/Output/path”
E.g. spark-submit Krish_Mehta_SON.py "/Users/krishmehta/Desktop/DataMining/Krish_Mehta_HW3/hw3/Data/yelp_reviews_large.txt" "100000" " Krish_Mehta_SON_yelp_reviews_large_100000.txt"

Approach:

Used apriori algorithm to implement the dataset with SON. In SON algorithm the input gets partitioned in chunks. For the phase 1, I have implemented Apriori algorithm for all the chunks. Continuously, filtered the datasets set with the support required until there are no more candidates available. In reduce phase, resolved the counts. In 2nd Phase of SON, the input is again partitioned into chunks and the occurrences of the candidate frequent itemset is counted and in reduce phase it adds up the counts and filters out only the ones above the threshold. Thus, obtaining the result.

The key use of apriori is that if the frequency of the of an itemset is less than the frequency of the superset item will also be less.

Bonus Question:

The higher the value of the threshold the lower will be the computation. Because as the threshold value will be high the number of valid candidate set reduces. The bottleneck for my threshold will be that if the value of threshold will be very low such that almost no itemset can be filtered then there will be a very high computation.

Yelp_Reviews_Test

Support Threshold	Execution Time
30	19.2347269058 seconds
40	5.17220497131 seconds

Yelp_Reviews_Small

Support Threshold	Execution Time
500	18.49117207553 seconds
1000	4.63750004768 seconds

Yelp_Reviews_Large

Support Threshold	Execution Time
100000	361.700501919 seconds
120000	294.586051941 seconds