TaskTwo

First, read GEO.txt and PatientMetaData.txt. For GEO.txt, filter out the header and the gene whose expression value is lower than 125000. For PatientMetaData.txt, filter out the header and the patients without cancer. Second, join GEO.txt and PatientMetaData.txt to have the cancer patients’ strong expression value gene stored in JavaPairRDD<String, String> “pid\_gid”. Use “pid\_gid” to calculate the minimum support value stored in “support\_num” and the maximum item-set size “k\_max”. To have the maximum number, a comparator class called “KMaxComparator” is defined by our group. Third, using “pid\_gid”, to have the size one item-set stored in JavaPairRDD<List<String>, List<String>> “gene\_set\_size\_one\_pid”. Key of “gene\_set\_size\_one\_pid” stores the gene set, by now, the list size is only one. Value of this stores all of the patients’ id who have contain all the genes that occurs in the corresponding key of “gene\_set\_size\_one\_pid”. Sort “gene\_set\_size\_one\_pid” with its key. To compare the List<String>, a comparator class called “ListComparator” is defined by our group. Fourth, prepare for the iteration. Use JavaPairRDD<List<String>, List<String>> “gene\_set\_pid\_occ” to store the final result. Let the iteration start from item-set size two. Have a flag to check if all of the item-sets have been filtered because of the support number. After these, start the iteration. Fifth, use the Apriori Algorithm to generate the next size of the item-set, stored in JavaPairRDD<List<String>, List<String>> “gene\_set\_size\_k\_pid”. Filter out the records that the size of the list (the list at the value situation of “gene\_set\_size\_k\_pid”) is smaller than the support number. If all of the candidate item-sets are filtered, the loop flag will be set to false. The iteration will stop. Sixth, convert “gene\_set\_pid\_occ” to the output format stored in JavaRDD<String> “output”, and save this “output” to the user defined output path.