**Users Topic Matching:(Java File)**

Recommending friends to users whom have high commonality in topics

package finalpreprocess1;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Collections;

import java.util.HashMap;

import java.util.HashSet;

import java.util.Iterator;

import java.util.List;

import java.util.Map;

import java.util.Scanner;

import java.util.Set;

import java.util.TreeMap;

public class topic {

static Map<String, String> friend\_list = new HashMap<String, String>();

public static void hashing() throws IOException

{

Map<String, Set<String>> map = new HashMap<String, Set<String>>();

// create list one and store values

//String users[]= {"us1","us2","us3","us4","us5","us6","us7","us8"};

Scanner sc=new Scanner(System.in);

System.out.println("Enter number of users\t");

int num=sc.nextInt();

String users[]=new String[num];

int x=0;

for(int h=0;h<num;h++)

{

Scanner scn=new Scanner(System.in);

System.out.println("Enter an user\t");

String filename=scn.next();

users[x++]=filename;

}

for(int i=0;i<users.length;i++)

{

try {

FileReader reader = new FileReader("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/"+users[i]+".txt");

BufferedReader bufferedReader = new BufferedReader(reader);

String topic="";

while ((topic = bufferedReader.readLine()) != null)

{

if(map.containsKey(topic))

{

Set l = map.get(topic);

l.add(users[i]);

map.put(topic, l);

}

else

{

Set l = new HashSet();

l.add(users[i]);

map.put(topic, l);

}

}

}

catch(IOException e)

{

e.printStackTrace();

}

}

//System.out.println("Fetching Keys and corresponding [Multiple] Values n");

try {

FileWriter writer = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/reverseindex.txt", true);

for (Map.Entry<String, Set<String>> entry : map.entrySet()) {

String k = entry.getKey();

Set<String> values = entry.getValue();

writer.write("Topic = "+k);

writer.write("\r\n"); // write new line

writer.write("Users = "+values.toString()+"\r\n");

writer.write("\r\n"); // write new line

}

writer.close();

}

catch (Exception e) {

e.printStackTrace();

}

try {

FileWriter writer = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/adjmatrix.txt", true);

writer.write("Matrix representation\r\n");

writer.write(" ");

for (Map.Entry<String, Set<String>> entry : map.entrySet()) {

String k = entry.getKey();

writer.write(k+" ");

}

writer.write("\r\n");

for(int j=0;j<users.length;j++)

{

writer.write(users[j]);//friend's list key

for (Map.Entry<String, Set<String>> entry : map.entrySet())

{

String k = entry.getKey();

Set a=map.get(k);

if(a.contains(users[j]))

{

writer.write("\t1");//friend's list value

String c=new String();

if(friend\_list.containsKey(users[j]))

c=friend\_list.get(users[j]);

friend\_list.put(users[j], c+"1");

}

else

{

writer.write("\t0");//friend's list value

String c=new String();

if(friend\_list.containsKey(users[j]))

c=friend\_list.get(users[j]);

friend\_list.put(users[j], c+"0");

}

}

writer.write("\r\n");

}

writer.close();

}

catch (Exception e) {

e.printStackTrace();

}

try

{

int matrix[][]=new int[10000][10000] ;

for(int j=0;j<users.length;j++)

{

for(int k=0;k<users.length;k++)

{

matrix[j][k]=0;

}

}

FileWriter fw = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/fmgraph.txt", true);

//System.out.println("Graph representation\r\n");

fw.write("Graph representation\r\n");

//System.out.println("Graph representation\r\n");

for(int j=0;j<users.length;j++)

{

for(int k=0;k<users.length;k++)

{

if(j==k)

{

}

else

{

int flag=0;

for (Map.Entry<String, Set<String>> entry : map.entrySet())

{

String p = entry.getKey();

Set<String> values = entry.getValue();

if(values.contains(users[j]) && values.contains(users[k]))

matrix[j][k]+=1;

}

/\*if(flag==1)

{

//if(matrix[k][j]==0)

//{

matrix[j][k]+=1;

System.out.println("j:"+j+"k:"+k);

// writer.write("1,");

/\*}

else

{

writer.write("0,");

}

}

else

{

// writer.write("0,");

}\*/

}

}

}

for(int j=0;j<users.length;j++)

{

for(int k=0;k<users.length;k++)

{

if(matrix[j][k]>3)

{

fw.write(matrix[j][k]+",");

}

else

{

matrix[j][k]=0;

fw.write(matrix[j][k]+",");

}

}

fw.write("\r\n");

}

fw.close();

}

catch (Exception e) {

e.printStackTrace();

}

}

/////////////////

public static void individual\_graph() throws IOException

{

Map<String, Set<String>> map = new HashMap<String, Set<String>>();

// create list one and store values

String users[]= {"ps","muthu","raj","david"};

for(int i=0;i<users.length;i++)

{

try {

FileReader reader = new FileReader("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/"+users[i]+".txt");

BufferedReader bufferedReader = new BufferedReader(reader);

String topic="";

while ((topic = bufferedReader.readLine()) != null)

{

if(map.containsKey(topic))

{

Set l = map.get(topic);

l.add(users[i]);

map.put(topic, l);

}

else

{

Set l = new HashSet();

l.add(users[i]);

map.put(topic, l);

}

}

}

catch(IOException e)

{

e.printStackTrace();

}

}

//System.out.println("Fetching Keys and corresponding [Multiple] Values n");

try {

FileWriter writer = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/split\_kv.txt", true);

for (Map.Entry<String, Set<String>> entry : map.entrySet()) {

String k = entry.getKey();

Set<String> values = entry.getValue();

writer.write("Topic = "+k);

writer.write("\r\n"); // write new line

writer.write("Users = "+values.toString()+"\r\n");

writer.write("\r\n"); // write new line

}

writer.close();

}

catch (Exception e) {

e.printStackTrace();

}

try {

FileWriter writer = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/split\_matrix.txt", true);

writer.write("Matrix representation\r\n");

writer.write(" ");

for (Map.Entry<String, Set<String>> entry : map.entrySet()) {

String k = entry.getKey();

writer.write(k+" ");

}

writer.write("\r\n");

for(int j=0;j<users.length;j++)

{

writer.write(users[j]);

for (Map.Entry<String, Set<String>> entry : map.entrySet())

{

String k = entry.getKey();

Set a=map.get(k);

if(a.contains(users[j]))

{

writer.write("\t1");

}

else

{

writer.write("\t0");

}

}

writer.write("\r\n");

}

writer.close();

}

catch (Exception e) {

e.printStackTrace();

}

try

{

FileWriter fw = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/split\_graph.txt", true);

//System.out.println("Graph representation\r\n");

fw.write("Graph representation\r\n");

//System.out.println("Graph representation\r\n");

for (Map.Entry<String, Set<String>> entry : map.entrySet())

{

String p = entry.getKey();

Set<String> values = entry.getValue();

int matrix[][]=new int[100][100] ;

for(int j=0;j<users.length;j++)

{

for(int k=0;k<users.length;k++)

{

matrix[j][k]=0;

}

}

for(int j=0;j<users.length;j++)

{

for(int k=0;k<users.length;k++)

{

if(j==k)

{

}

else

{

int flag=0;

if(values.contains(users[j]) && values.contains(users[k]))

matrix[j][k]=1;

}

}

}

fw.write("Topic Name:"+p+"\r\n");

for(int j=0;j<users.length;j++)

{

for(int k=0;k<users.length;k++)

{

fw.write(matrix[j][k]+",");

}

fw.write("\r\n");

}

fw.write("\r\n");

}

fw.close();

}

catch (Exception e) {

e.printStackTrace();

}

}

public static void friend\_suggestion() throws IOException

{

try

{

System.out.println("\nFriend suggestions for each user\n");

for (Map.Entry<String, String> entry : friend\_list.entrySet())

{

String k = entry.getKey();

String topics1=friend\_list.get(k);//System.out.println(topics1);

Map<Integer, Set<String>> common\_list = new TreeMap<Integer, Set<String>>(Collections.reverseOrder());

for (Map.Entry<String, String> enter : friend\_list.entrySet())

{

String e = enter.getKey();

if(!e.equals(k))

{//System.out.println(k+" "+e);

String topics2=friend\_list.get(e);

int common=0;

for(int p=0;p<topics1.length();p++)//user1.length()==user2.length()

{

if(topics1.charAt(p)=='1' && topics2.charAt(p)=='1')

{

common++;

}

}//System.out.println(k+e+"common= "+common);

if(!common\_list.containsKey(common))

{

Set a=new HashSet();

a.add(e);

common\_list.put(common,a);

}

else

{

Set b=common\_list.get(common);

b.add(e);

common\_list.put(common,b);

}

}

}

System.out.println(k);

int top\_friends=0;

FileWriter fw = new FileWriter("C:/Users/PANNEER SELVAM/Downloads/Desktop/finalpreprocess/finalpreprocess1/FRIEND\_SUGGESTION.txt", true);

for (Map.Entry<Integer, Set<String>> common\_entry : common\_list.entrySet())

{

Integer y = common\_entry.getKey();

Set friends=common\_list.get(y);

ArrayList friend=new ArrayList(friends);

Iterator<String> iterator = friend.iterator();

while (iterator.hasNext())

{

if(top\_friends++<5)

{

System.out.print(top\_friends);

System.out.print(" "+iterator.next()+"\n");

//fw.write(top\_friends);

//fw.write(" "+iterator.next()+"\r\n\r\n");

}

else

break;

}

}

fw.close();

System.out.println("\n");

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

public static void main(String args[]) throws IOException

{

hashing();

friend\_suggestion();

// individual\_graph();

}

}