Tsz Yan Wong

CSE 4701

**(i) Source Code (.java file is submitted separately on HuskyCT)  
  
\* Results and tables for A and B are on page 5**  
  
/\*

 \* Name: Tsz Yan Wong

 \* CSE 4701 Database - Project 2 Part 2

 \*/

package Project\_2\_2;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

public class Project\_2\_2 {

    public static void main(String args[]) {

        ArrayList<String> Mutations = new ArrayList<String>();

    Mutations.add("APC");

    Mutations.add("TP53");

    Mutations.add("KRAS");

    Mutations.add("PIK3CA");

    Mutations.add("PTEN");

    Mutations.add("ATM");

    Mutations.add("MUC4");

    Mutations.add("SMAD4");

    Mutations.add("SYNE1");

    Mutations.add("FBXW7");

        Connection con;

        Statement stmt;

        ResultSet rs;

        ResultSetMetaData rsmd;

        /\* Database credentials \*/

        String user = "cse4701";

        String password = "datamine";

        String host = "query.engr.uconn.edu";

        String port = "1521";

        String sid = "BIBCI";

        String url = "jdbc:oracle:thin:@" + host + ":" + port + ":" + sid;

        double A\_Y1 = 0;

        double B\_Y0 = 0;

        double C\_N1 = 0;

        double D\_N0;

        double infoNeed;

        double[] APC = new double[1];

        double[] TP53 = new double[1];

        double[] KRAS = new double[1];

        double[] PIK3CA = new double[1];

        double[] PTEN = new double[1];

        double[] ATM = new double[1];

        double[] MUC4 = new double[1];

        double[] SMAD4 = new double[1];

        double[] SYNE1 = new double[1];

        double[] FBXW7 = new double[1];

        double[][] arrays = new double[][] {APC, TP53, KRAS, PIK3CA, PTEN, ATM, MUC4, SMAD4, SYNE1, FBXW7};

        double ig;

        //Create two new arrays for the final result

        double[] gains = new double[10]; // Store the IGs for the 10 genes in descending order (later after the try block)

        int[] overlap\_A = new int[10];

        try {

            // Make a connection to the Oracle Database

            DriverManager.registerDriver(new oracle.jdbc.OracleDriver());

            con = DriverManager.getConnection(url, user, password);

            stmt = con.createStatement();

            /\*\*

             \* Dead = 497, Alive = 130

             \* Entropy is calculated by Info(D) = I(130, 497)

             \*/

            double original\_entropy = compute\_entropy(130, 497);

            //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

            // Get A,B,C,D for all the genes

            for(int i = 0; i < Mutations.size(); i++) {

                String sql = "SELECT COUNT(" + Mutations.get(i) + ") FROM IG\_READY WHERE STATUS = 'Y' AND " + Mutations.get(i) + " = 1";

                rs = stmt.executeQuery(sql);

                rsmd = rs.getMetaData();

                while(rs.next()) {

                    A\_Y1 = rs.getDouble(1);

                }

                overlap\_A[i] = (int)A\_Y1;

                sql = "SELECT COUNT(" + Mutations.get(i) + ") FROM IG\_READY WHERE STATUS = 'Y' AND " + Mutations.get(i) + " != 1";

                rs = stmt.executeQuery(sql);

                rsmd = rs.getMetaData();

                while(rs.next()) {

                    B\_Y0 = rs.getDouble(1);

                }

                sql = "SELECT COUNT(" + Mutations.get(i) + ") FROM IG\_READY WHERE STATUS = 'N' AND " + Mutations.get(i) + " = 1";

                rs = stmt.executeQuery(sql);

                rsmd = rs.getMetaData();

                while(rs.next()) {

                    C\_N1 = rs.getDouble(1);

                }

                D\_N0 = 627 - (A\_Y1 + B\_Y0 + C\_N1);

                infoNeed = compute\_infoD(A\_Y1, B\_Y0, C\_N1, D\_N0);

                ig = info\_gain(original\_entropy, infoNeed);

                arrays[i][0] = ig;

                gains[i] = ig; //Store the IGs in another array

                System.out.println("The Information Gain for " + Mutations.get(i) + " is " + arrays[i][0]);

            }

        }

        catch (SQLException ex) {

            System.out.println("Error Occured");

        }

        String[] genes = {"APC", "TP53", "KRAS", "PIK3CA", "PTEN", "ATM", "MUC4", "SMAD4", "SYNE1", "FBXW7"};

        //Using Selection Sort to sort the gains array as well as the genes array

        int minIndex;

        double minValue;

        String minString;

        int minOcnt;

        int index;

        for(int startScan = 0; startScan < gains.length - 1; startScan++) {

            minIndex = startScan;

            minValue = gains[startScan];

            minString = genes[startScan];

            minOcnt = overlap\_A[startScan];

            for(index = startScan + 1; index < gains.length; index++) {

                if(gains[index] < minValue) {

                    minValue = gains[index]; //For the gains array

                    minString = genes[index]; //For the genes array

                    minOcnt = overlap\_A[index]; //For O CNT

                    minIndex = index;

                }

            }

            gains[minIndex] = gains[startScan];

            gains[startScan] = minValue;

            genes[minIndex] = genes[startScan];

            genes[startScan] = minString;

            overlap\_A[minIndex] = overlap\_A[startScan];

            overlap\_A[startScan] = minOcnt;

        }

        //Print out the data, top 5 information gain ranked

        System.out.println();

        System.out.println(" ------------------------------------------------------");

        System.out.println("|      Gene ID    |        IG        |       O CNT     |");

        System.out.println(" ------------------------------------------------------");

        //Data

        System.out.printf("|      %6s     |    %.8f    |     %5d       |\n", genes[9], gains[9], overlap\_A[9]);

        System.out.println(" ------------------------------------------------------");

        System.out.printf("|      %6s     |    %.8f    |     %5d       |\n", genes[8], gains[8], overlap\_A[8]);

        System.out.println(" ------------------------------------------------------");

        System.out.printf("|      %6s     |    %.8f    |     %5d       |\n", genes[7], gains[7], overlap\_A[7]);

        System.out.println(" ------------------------------------------------------");

        System.out.printf("|      %6s     |    %.8f    |     %5d       |\n", genes[6], gains[6], overlap\_A[6]);

        System.out.println(" ------------------------------------------------------");

        System.out.printf("|      %6s     |    %.8f    |     %5d       |\n", genes[5], gains[5], overlap\_A[5]);

        System.out.println(" ------------------------------------------------------");

        System.out.println();

        System.out.println("The End");

        System.out.println();

        System.exit(0);

    }

    /\* A method that calculates the information gain and return it for ranking \*/

    public static double info\_gain(double entropy, double info\_needed) {

        // calculate information gain

        return entropy - info\_needed;

    }

    /\* A method that computes the entropy \*/

    public static double compute\_entropy(double x, double y) {

        double entropy;

        // compute the entropy

        double x1 = x / (x+y);

        double y1 = y / (x+y);

        // logb(n) = loge(n) / loge(b)

        entropy = -x1 \* (Math.log(x1)/Math.log(2)) - (y1 \* (Math.log(y1)/Math.log(2)));

        return entropy;

    }

    /\* A method that computes the Information needed, aka InfoA(D) \*/

    public static double compute\_infoD(double A, double B, double C, double D) {

        /\*\*

         \* A = Dead with mutation

         \* B = Dead with no mutation

         \* C = Alive with mutation

         \* D = Alive with no mutation

         \*/

        // Modify the data if they are 0 so that the NaN situation doesn't occur

        if(A==0)

            A = 0.00000000001;

        if(B==0)

            B = 0.00000000001;

        if(C==0)

            C = 0.00000000001;

        if(D==0)

            D = 0.00000000001;

        double infoD\_mutation;

        double infoD\_noMutation;

        double mutation = A + C;

        double no\_mutation = B + D;

        double total = mutation + no\_mutation;

        infoD\_mutation = (mutation/total) \* compute\_entropy(A, C);

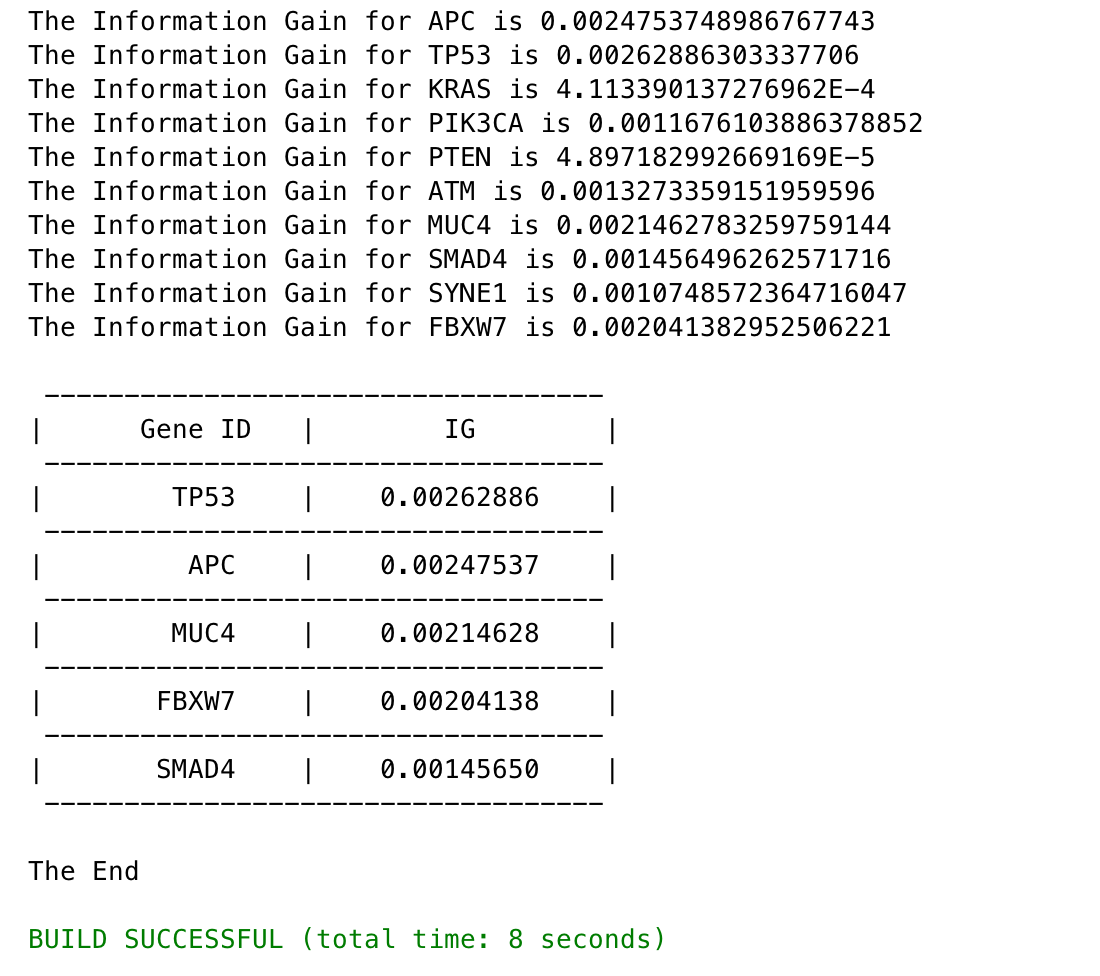
        infoD\_noMutation = (no\_mutation/total) \* compute\_entropy(B, D);

        return infoD\_mutation + infoD\_noMutation;

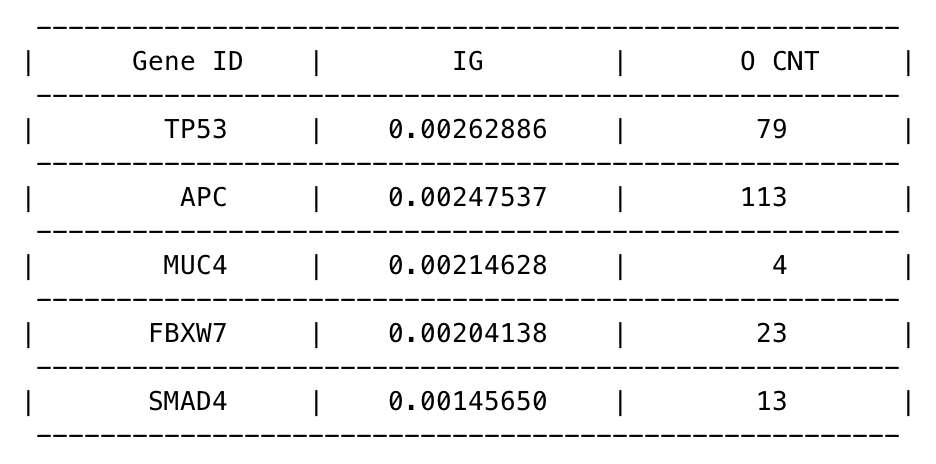
    }

}

**(ii) Result & Table for A**



**(iii) Table for B**



**(iv) Write-up**

Based on the table I generated in (iii), the ordering based on IG and O CNT does not coincide. The information gain for TP53 is 0.00262886, which is the highest among the genes, whereas the O CNT for TP53 is not the highest. One reason that they do not coincide might be that IG takes “false negative” and “false positive” situations into account, whereas O CNT only cares about the “true positive” situations. So, in conclusion, more overlapping does not mean higher information gain and vice versa.