Name: Kushal Kishor Shankhapal

Group B, Assignment 4-5: Precision, Recal, Harmonic mean (F-measure) and E-measure

Problem Statement:

Implement a program to calculate precision and recall for sample input.
(Answer set A, Query q1, Relevant documents to query q1- Rq1)
Write a program to calculate harmonic mean (F-measure) and E-measure for above example

Recall_Precision_Evaluator.cpp

```
#include <iostream>
#include <string.h>
#include <iomanip>
#include <fstream>

using namespace std;

string left(const string s, const int w)
{ // Left aligns input string in table
    stringstream ss, spaces;
    int padding = w - s.size(); // count excess room to pad
    for (int i = 0; i < padding; ++i)
        spaces << " ";
    ss << s << spaces.str() << '|'; // format with padding
    return ss.str();
}

string center(const string s, const int w)
{ // center aligns input string in table
    stringstream ss, spaces;
    int padding = w - s.size(); // count excess room to pad
    for (int i = 0; i < padding / 2; ++i)
        spaces << " ";
    ss << spaces.str() << s << spaces.str(); // format with padding
    if (padding > 0 && padding % 2 != 0)     // if odd #, add 1 space
        ss << " ";
    return ss.str();
}

string prd(float x, int decDigits, int width)
{ // right aligns float values with specified no. of precision digits in a table
    stringstream ss;
    ss << fixed << right;
    ss.fill(' ');            // fill space around displayed #
    ss.width(width);         // set  width around displayed #
    ss.precision(decDigits); // set # places after decimal
    ss << x;
    return ss.str();
}

string printDocs(string state[], int size)
{
    // prints each document at a specific iteration inside the table
    stringstream ss;
    ss << '|' << ' ';
    for (int i = 0; i < size; i++)
    { // convert the array into a string of comma seprated values
        ss << state[i];
        if (state[i].compare("") != 0 and i + 1 < size and state[i + 1].compare("") != 0)
            ss << ',' << ' ';
    }
    return left(ss.str(), 98);
}
```

```
float E_value(float b, float rj, float pj)
{ // calculates E value
    return 1 - (((1 + b * b) * rj * pj) / (b * b * pj + rj));
}

int main()
{ // Hardcoded Rq and A
    string Rq[10] = {"d3", "d5", "d9", "d25", "d39", "d44", "d56", "d71", "d89", "d123"};
    string A[15] = {"d123", "d84", "d56", "d6", "d8", "d9", "d511", "d129", "d187",
"d25", "d38", "d48", "d250", "d113", "d3"};

    // Creating and opening output file
    ofstream write("Recall_Precision_Evaluation_output.txt");

    // required constants and arrays for calculations
    float modRq = sizeof(Rq) / sizeof(Rq[0]);
    string Ra[sizeof(A) / sizeof(A[0])];
    float P[sizeof(A) / sizeof(A[0])];
    float R[sizeof(A) / sizeof(A[0])];
    float modRa = 0;
    float modA = 0;
    double precision;
    double recall;

    // table header formatting and printing
    std::cout << setprecision(2) << fixed;
    write << setprecision(2) << fixed;
    std::cout << string(45 * 3 + 11, '-') << "\n";
    write << string(45 * 3 + 11, '-') << "\n";
    std::cout << '|' << center("Documents", 96) << " | "
              << center("|Ra|", 8) << " | "
              << center("|A|", 8) << " | "
              << center("Precision(%)", 5) << "|"
              << center("Recall(%)", 5) << " | " << endl;
    write << '|' << center("Documents", 96) << " | "
          << center("|Ra|", 8) << " | "
          << center("|A|", 8) << " | "
          << center("Precision(%)", 5) << "|"
          << center("Recall(%)", 5) << " | " << endl;
    std::cout << string(45 * 3 + 11, '-') << "\n";
    write << string(45 * 3 + 11, '-') << "\n";
    // Algorithm to calculate and print all the values in the output table, MAIN algo
    for (int i = 0; i < sizeof(A) / sizeof(A[0]); i++)
    {
        Ra[i] = A[i];
        modA++;
        for (int j = 0; j < modRq; j++)
        {
            if (A[i] == Rq[j])
            {
                modRa++;
                break;
            }
        }
        precision = (modRa / modA) * 100;
        P[i] = precision / 100;
        recall = (modRa / modRq) * 100;
        R[i] = recall / 100;
        // Printing documents and other values of current iteration within the table
        std::cout << printDocs(Ra, sizeof(Ra) / sizeof(Ra[0]));
        write << printDocs(Ra, sizeof(Ra) / sizeof(Ra[0]));
        std::cout << prd(modRa, 2, 10) << "|"
                  << prd(modA, 2, 10) << "|"
                  << prd(precision, 2, 13) << "|"
                  << prd(recall, 2, 10) << "|"
                  << endl;
        write << prd(modRa, 2, 10) << "|"
              << prd(modA, 2, 10) << "|"
              << prd(precision, 2, 13) << "|"
              << prd(recall, 2, 10) << "|"
              << endl;
    }
    // closing the table
    std::cout << string(45 * 3 + 11, '-') << "\n";
    write << string(45 * 3 + 11, '-') << "\n";

    // taking user input for calculation of Fj and Ej
    int j;
    do
    {
        std::cout << "Harmonic mean and E-value\nEnter value of j(0 - " << (sizeof(A) /
sizeof(A[0])) - 1 << ") to find F(j)and E(j):" << endl;
        cin >> j;
    } while (j > sizeof(Ra) / sizeof(Ra[0]));

    // calculating Harmonic mean and printing in table
    float Fj = (2 * P[j] * R[j]) / (P[j] + R[j]);
    std::cout << string(15 * 2 + 3, '-') << "\n"
              << "| Harmonic mean (F" << j << ") is: |" << Fj << " |\n"
              << string(15 * 2 + 3, '-') << "\n";
    write << string(15 * 2 + 3, '-') << "\n"
          << "| Harmonic mean (F" << j << ") is: |" << Fj << " |\n"
          << string(15 * 2 + 3, '-') << "\n";

    // table header
    std::cout << string(15 * 2 + 4, '-') << "\n"
              << "|" << center("E-Value", 32) << "|\n"
              << string(15 * 2 + 4, '-') << "\n";
    write << string(15 * 2 + 4, '-') << "\n"
          << "|" << center("E-Value", 32) << "|\n"
          << string(15 * 2 + 4, '-') << "\n";

    // table header (sub columns)
    std::cout << "|" << center("b>1", 10) << "|"
              << center("b=0", 10) << "|"
              << center("b<1", 10) << "|\n"
              << string(15 * 2 + 4, '-') << "\n";
    write << "|" << center("b>1", 10) << "|"
          << center("b=0", 10) << "|"
          << center("b<1", 10) << "|\n"
          << string(15 * 2 + 4, '-') << "\n";

    // Calculating and Printing E-Values in table
    std::cout << "|" << prd(E_value(1.1, R[j], P[j]), 2, 10) << "|"
              << prd(E_value(0, R[j], P[j]), 2, 10) << "|"
              << prd(E_value(0.9, R[j], P[j]), 2, 10) << "|\n";
    write << "|" << prd(E_value(1.1, R[j], P[j]), 2, 10) << "|"
          << prd(E_value(0, R[j], P[j]), 2, 10) << "|"
          << prd(E_value(0.9, R[j], P[j]), 2, 10) << "|\n";
    // Closing table
    std::cout << string(15 * 2 + 4, '-') << "\n";
    write << string(15 * 2 + 4, '-') << "\n";
    write.close();
    return 0;
}
```

Output:

```
it@IT-SWL-19:~$ g++ Recall_Precision_Evaluator.cpp
it@IT-SWL-19:~$ ./a.out
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
|                                                         Documents
|    |Ra|    |    |A|     | Precision(%)|Recall(%) |
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
| d123
|       1.00|       1.00|       100.00|     10.00|
| d123, d84
|       1.00|       2.00|        50.00|     10.00|
| d123, d84, d56
|       2.00|       3.00|        66.67|     20.00|
| d123, d84, d56, d6
|       2.00|       4.00|        50.00|     20.00|
| d123, d84, d56, d6, d8
|       2.00|       5.00|        40.00|     20.00|
| d123, d84, d56, d6, d8, d9
|       3.00|       6.00|        50.00|     30.00|
| d123, d84, d56, d6, d8, d9, d511
|       3.00|       7.00|        42.86|     30.00|
| d123, d84, d56, d6, d8, d9, d511, d129
|       3.00|       8.00|        37.50|     30.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187
|       3.00|       9.00|        33.33|     30.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25
|       4.00|      10.00|        40.00|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38
|       4.00|      11.00|        36.36|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48
|       4.00|      12.00|        33.33|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48, d250
|       4.00|      13.00|        30.77|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48, d250, d113
|       4.00|      14.00|        28.57|     40.00|
| d123, d84, d56, d6, d8, d9, d511, d129, d187, d25, d38, d48, d250, d113, d3
|       5.00|      15.00|        33.33|     50.00|
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Harmonic mean and E-value
Enter value of j(0 - 14) to find F(j)and E(j):
7
---------------------------------
| Harmonic mean (F7) is: |0.33 |
---------------------------------
----------------------------------
|            E-Value             |
----------------------------------
|   b>1    |   b=0    |   b<1    |
----------------------------------
|      0.67|      0.62|      0.66|
----------------------------------
it@IT-SWL-19:~$
```