Experiment no. 11

Name: Sonali Dattatray Kaingade

PRN: 21620002

Title: Write a program for Agglomerative Hierarchical clustering using single linkage method

code:

```
#include <bits/stdc++.h>
using namespace std;

int op = 1;

ofstream fwtr("linkage_output.csv", ios::out);

// Function to perform agglomerative clustering and return the name of the resulting cluster
string agglomerative(string input)
{
   map<string, map<string, int>> dm;

   fstream file(input, ios::in);

   string line;
   getline(file, line);

   int pt = 0;
   stringstream st(line);
```

```
int i = 0;
string point;
vector<string> points;

// Read the point names from the first line of the input file
while (getline(st, point, ','))
{
   if (i == 0)
   {
      i++;
      continue;
   }
   points.push_back(point);
}

// Populate the distance matrix from the input file
while (getline(file, line))
{
   stringstream str(line);
   getline(str, point, ',');

   string dist;
   int idx = 0;

   while (getline(str, dist, ','))
```

```
    {
        if (dist.length() != 0)
            dm[point][points[idx]] = stoi(dist);

        idx++;
    }
}

string pt1, pt2;
int min_dist = INT_MAX;

// Find the two points with the minimum distance
for (auto p : dm)
{
    for (auto pp : p.second)
    {
        string p1 = p.first, p2 = pp.first;
        int dist = pp.second;

        if (p1 != p2 && dist < min_dist)
        {
            pt1 = p1;
            pt2 = p2;
            min_dist = dist;
        }
```

```
    }
}

cout << "Clusters Chosen: " << pt1 << " & " << pt2 << endl;

string up, down;

// Determine the order of the two points based on their names
if (pt1[0] > pt2[0])
{
    up = pt2;
    down = pt1;
}
else
{
    up = pt1;
    down = pt2;
}

string newPt = down + up;

// Update distances and remove old points from the matrix
for (auto p : dm)
{
    point = p.first;
```

```
    if (point[0] > newPt[0])
    {
        dm[point][newPt] = min(dm[point][up], dm[point][down]);
    }
}

for (auto p : dm[down])
{
    point = p.first;
    int d1 = p.second;

    if (point[0] < up[0])
        d1 = min(d1, dm[up][point]);
    else
        d1 = min(d1, dm[point][up]);

    dm[newPt][point] = d1;
}

for (auto p : dm)
{
    point = p.first;
    auto mtemp = p.second;
```

```
    if (point[0] >= up[0])
    {
        int d1 = dm[point][up];

        if (down[0] > point[0])
            d1 = min(d1, dm[down][point]);
        else
            d1 = min(d1, dm[point][down]);

        dm[point][newPt] = d1;
        dm[point].erase(up);

        if (point[0] >= down[0])
            dm[point].erase(down);
    }
}

dm.erase(up);
dm.erase(down);

// Create an output file with updated cluster data
string output = "output" + to_string(op++) + ".csv";
ofstream fw(output, ios::out);
fw << ",";
```

```
    for (auto p : dm)
    {
        fw << p.first << ",";
    }
    fw << "\n";

    for (auto p : dm)
    {
        fw << p.first << ",";
        for (auto pp : p.second)
        {
            fw << pp.second << ",";
        }
        fw << "\n";
    }

    fw.close();
    fwtr << down << " & " << up << "\n";

    return output;
}

int main()
{
    string input = "linkage_input.csv";
```

```
fstream file1(input, ios::in);

string line;
getline(file1, line);

int pt = 0;
stringstream st(line);

int j = 0, len = 0;
string point;

// Determine the number of points in the dataset
while (getline(st, point, ','))
{
   if (j == 0)
   {
      j++;
      continue;
   }
   len++;
}

// Repeatedly perform agglomerative clustering to create clusters
for (int i = 1; i <= len - 2; i++)
```

```
    {
        string output = agglomerative(input);
        input = output;
    }
    return 0;
}
```

Result:

Input.csv:

	A	B	C	D	E	F
A	0					
B	16	0				
C	47	37	0			
D	72	57	40	0		
E	77	65	30	31	0	
F	79	66	35	23	10	0

Output.csv:

1.

	A	B	C	D	FE
A	0				
B	16	0			
C	47	37	0		
D	72	57	40	0	
FE	77	65	30	23	0

2.

	BA	C	D	FE
BA	0			
C	37	0		
D	57	40	0	
FE	65	30	23	0

3.

	BA	C	FED
BA	0		
C	37	0	
FED	57	30	0

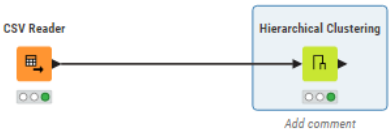
4.

	BA	FEDC
BA	0	
FEDC	37	0

final output:

F & E
B & A
FE & D
FED & C

knime:



▶ 1: Clustered data Flow Variables

Rows: 6 | Columns: 8 Table Statistics

#	Row...	Column0 String	A Number (inte...	B Number (inte...	C Number (inte...	D Number (inte...	E Number (inte...	F Number (inte...	Cluster String
1	Row2	C	47	37	0	?	?	?	cluster_0
2	Row0	A	0	?	?	?	?	?	cluster_1
3	Row1	B	16	0	?	?	?	?	cluster_1
4	Row3	D	72	57	40	0	?	?	cluster_2
5	Row4	E	77	65	30	31	0	?	cluster_2
6	Row5	F	79	66	35	23	10	0	cluster_2