TUGAS M7
KNOWLEDGE DISCOVERY
CLUSTERING OPTIMIZATION

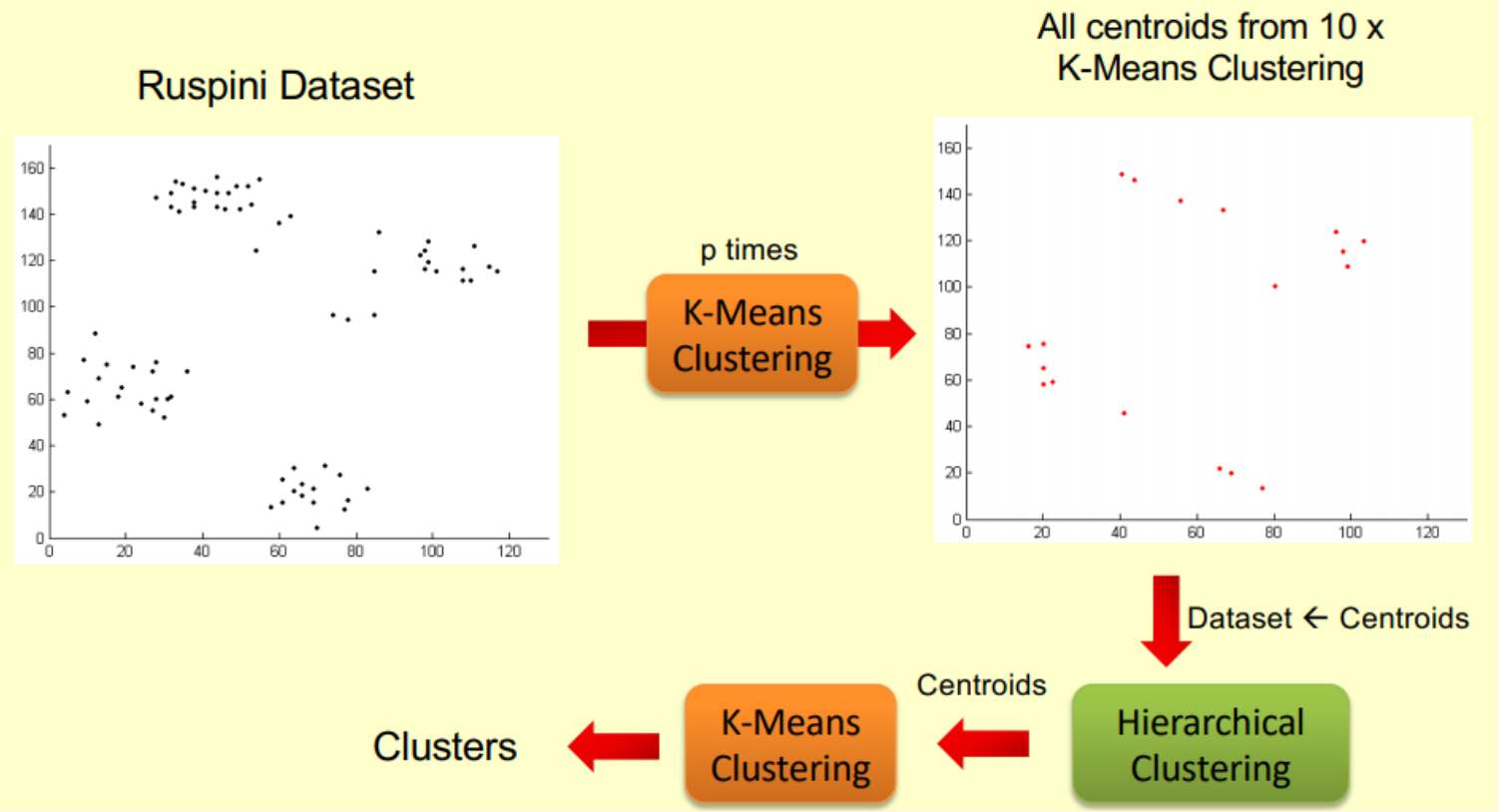
Ni Putu Devira AM
1120800012
S2 Elektro 2020

SOAL!

Assignment

1. Convert that data into the numerical values
2. Impute the missing data with the mean values of same attribute in the same class
3. Hide the class label of the supervised data
4. Normalize the data
5. Cluster the data using Hierarchical K-Means into 2 groups
6. Do cluster analysis and compare the performance with Hierarchical Clustering
7. How do you interpret the clusters which indicates LIVE and DIE?
8. Make the report and upload to the website (M7)

PROSES CLUSTERING OPTIMIZATION



LANGKAH PENGERJAAN

1. Inisialisasi Variabel

```
int i,j,k,l,b,a,ranking[200][200],iterasi,loop,newmax,newmin,cl,iterasicl;
float I[200][200],c[200][20],jarak[200],rank[200],jumlah[200],meanl[200][200],mean2[200][200],mean3[200][200], mean[200][200],kelas[200],
float varian[200];
```

2. Menginputkan Dataset

```
freopen("taskl.txt","r",stdin);//if You want to notepad , with name input.txt
for( int i = 1 ; i <=155; i++ )
{
    for(j=1; j<=19; j++)
    {
        scanf("%f", &I[i][j]);
    }
    scanf("%f",&kelas[i]);
}
```

LANGKAH PENGERJAAN

3. Normalisasi Dataset menggunakan Metode Normalisasi Min-Max (0-1)

```
//Data Baru yang sudah diNormalisasi Min-Max(0-1)
//Data Learning
newmax=1;
newmin=0;
for(i=1; i<=155; i++)
{
    I[i][1]=((I[i][1]-7)*(newmax-newmin))/((78-7)+newmin);
    I[i][14]=((I[i][14]-1)*(newmax-newmin))/((8-1)+newmin);
    I[i][15]=((I[i][15]-26)*(newmax-newmin))/((295-26)+newmin);
    I[i][16]=((I[i][16]-14)*(newmax-newmin))/((648-14)+newmin);
    I[i][17]=((I[i][17]-3)*(newmax-newmin))/((5-3)+newmin);
    I[i][18]=((I[i][18]-0)*(newmax-newmin))/((100-0)+newmin);
    //printf("%d = %f %f %f %f %f %f\n",i,Age[i],Bilir[i],Alk[i],Sgot[i],Albumin[i],Protime[i]);
}
```

LANGKAH PENGERJAAN

4. Merandom nilai Centroid

```
for(i=1; i<=cl; i++)
{
    for(j=1; j<=19; j++)
    {
        c[i][j]=rand()%100/99.1938129;
        //printf("c[%d][%d]= %f\n",i,j,c[i][j]);
    }
}
```

5. Mengiterasi proses clustering menggunakan metode K-Means

```
loop=10;
for(iterasi=1; iterasi<=loop; iterasi++)
{
```

LANGKAH PENGERJAAN

6. Menghitung jarak antar Cluster

```
    printf("\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\");
    printf("\niterasi = %d\n\n",iterasi);
for(i=1; i<=155; i++)
{
    for(j=1; j<=cl; j++) //CLuster
    {
        jarak[j]=0;
        for(k=1; k<=19; k++) //Penjumlahan jarak
        {
            jarak[j]=jarak[j]+pow(I[i][k]-c[j][k],2);
        }
        jarak[j]=sqrt(jarak[j]);
        //printf("jarak[%d][%d]=%f\n",i,j,jarak[j]);
    }
```

LANGKAH PENGERJAAN

7. Meranking nilai jarak Cluster dan memilih nilai yang paling kecil

```
for(j=1; j<=cl; j++)
    {
        rank[j]=1;
        for(b=1; b<=cl; b++)
            {
                if(jarak[j]>jarak[b])
                {
                    rank[j]++;
                }
            }
    }
    //printf("Nilai Ujian\t Rangking\n");
```

```
for(j=1; j<=cl; j++)
{
    for(k=1; k<=cl; k++)
    {
        if(rank[j]==1 && j==k)
        {
            jumlah[j]=jumlah[j]+1;
            ranking[i][j]=j;
        }
        else if(rank[j]!=1)
        {
            ranking[i][j]=0;
        }
    }
}
```

LANGKAH PENGERJAAN

7. Menghitung Jumlah data dari setiap attribute sebagai bagian dari proses mencari Mean

```
for(i=1; i<=155; i++)
{
    //printf("i= %d   ",i);
    for(j=1; j<=cl; j++)
    {
        for(k=1; k<=cl; k++)
        {
            if(ranking[i][j]==k)
            {
                kelastesting[i]=ranking[i][j];
                //printf("class[%d] = %f\n",i,kelastesting[i]);
                mean[ranking[i][j]][1]=mean[ranking[i][j]][1]+I[i][1];
                mean[ranking[i][j]][2]=mean[ranking[i][j]][2]+I[i][2];
                mean[ranking[i][j]][3]=mean[ranking[i][j]][3]+I[i][3];
                mean[ranking[i][j]][4]=mean[ranking[i][j]][4]+I[i][4];
                mean[ranking[i][j]][5]=mean[ranking[i][j]][5]+I[i][5];
                mean[ranking[i][j]][6]=mean[ranking[i][j]][6]+I[i][6];
                mean[ranking[i][j]][7]=mean[ranking[i][j]][7]+I[i][7];
                mean[ranking[i][j]][8]=mean[ranking[i][j]][8]+I[i][8];
                mean[ranking[i][j]][9]=mean[ranking[i][j]][9]+I[i][9];
                mean[ranking[i][j]][10]=mean[ranking[i][j]][10]+I[i][10];
                mean[ranking[i][j]][11]=mean[ranking[i][j]][11]+I[i][11];
                mean[ranking[i][j]][12]=mean[ranking[i][j]][12]+I[i][12];
                mean[ranking[i][j]][13]=mean[ranking[i][j]][13]+I[i][13];
                mean[ranking[i][j]][14]=mean[ranking[i][j]][14]+I[i][14];
                mean[ranking[i][j]][15]=mean[ranking[i][j]][15]+I[i][15];
                mean[ranking[i][j]][16]=mean[ranking[i][j]][16]+I[i][16];
                mean[ranking[i][j]][17]=mean[ranking[i][j]][17]+I[i][17];
                mean[ranking[i][j]][18]=mean[ranking[i][j]][18]+I[i][18];
                mean[ranking[i][j]][19]=mean[ranking[i][j]][19]+I[i][19];
            }
        }
    }
}
```

LANGKAH PENGERJAAN

8. Menghitung banyaknya data yang termasuk dalam cluster tertentu sebagai bagian dari proses mencari Mean

```
for(i=1; i<=cl; i++)
{
    for(k=1; k<=cl; k++)
    {
        if(i==k)
        {
            for(j=1; j<=19; j++)
            {
                if(jumlah[i]==0)
                {
                    c[i][j]=0;
                }
                else if(jumlah[i]!=0)
                {
                    c[i][j]=mean[i][j]/jumlah[i];
                }
                //printf("c[%d][%d] = %f\n",i,j,c[i][j]);
            }
        }
    }

}
```

LANGKAH PENGERJAAN

9. Mendapatkan alamat Centroid

```
if(iterasi!=1)
{
    for(i=1; i<=19; i++)
    {
        c[(iterasi*2)-1][i]=c[1][i];
        c[iterasi*2][i]=c[2][i];
        printf("c[%d][%d]= %f\n",(iterasi*2)-1,i,c[(iterasi*2)-1][i]);
        printf("c[%d][%d]= %f\n",(iterasi*2),i,c[iterasi*2][i]);
    }
}
```

```
for(i=1; i<=20; i++)
{
    for(j=1; j<=20; j++)
    {
        printf("c[%d][%d] = %f\n",i,j,c[i][j]);
    }
}
```

MENJADIKAN CENTROID YANG TELAH DIDAPATKAN, MENJADI DATASET BARU, YANG NANTINYA MENJADI INPUT DARI HIERARCHICAL CLUSTERING

LANGKAH PENGERJAAN

10. Menghitung jarak tiap Centroid

```
for(i=1; i<=20; i++)
{
    for(j=1; j<=20; j++)
    {
        dd[i][j]=0;
        if(i>j)
        {
            for(k=1; k<=19; k++) //Attribut
            {
               dd[i][j]=dd[i][j]+pow((c[i][k]-c[j][k]),2);
            }
            dd[i][j]=sqrt(dd[i][j]);
            printf("d[%d][%d] = %f\n",i,j,dd[i][j]);
        }
        else
        {
            dd[i][j]=10000;
            printf("d[%d][%d] = %f\n",i,j,dd[i][j]);
        }
    }
}
```

LANGKAH PENGERJAAN

11. Memilih ranking terbesar atau jarak centroid terkecil

```
looping=16;
for(iterate=1; iterate<=looping; iterate++)
{
    printf("\n\n\\\\\\\\\\\\\\%d \\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\ \n\n",iterate);
```

```
for(j=1; j<=20; j++)
        {
            for(k=1; k<=20; k++)
            {
                peranking[j][k]=1;
                for(b=1; b<=20; b++)
                    {
                        for(e=1; e<=20; e++)
                        {
                            {
                                if(dd[j][k]>dd[b][e])
                                {
                                    peranking[j][k]++;
                                }
                            }
                        }
                    }
            }
        }
```

LANGKAH PENGERJAAN

12. Proses mengubah nilai jarak =1000 menjadi jarak=0, agar mudah untuk diprogram (Manipulasi Program)

```
//Mengnolkan sisi kanan matriks
printf("\n");
for(i=1; i<=20; i++)
{
    for(j=1; j<=20; j++)
    {
        if(dd[i][j]==10000)
        {
            dd[i][j]=0;

        }
        printf("dd[%d][%d] = %f\n",i,j,dd[i][j]);
    }
}
```

LANGKAH PENGERJAAN

13. Menghitung nilai terbesar dari jarak terkecil antar centroid dari proses sebelumnya, menggunakan metode Complete Linkage

```
for(i=1; i<=20; i++)
{
    for(j=1; j<=20; j++)
    {
        if(peranking[i][j]==1)
        {
            for(k=1; k<=20; k++)
            {
                    if(dd[i][k]>=dd[j][k]) //complete linkage
                    {
                        dd[k][i]=dd[i][k]; //ditaruh di kolom terakhir
                    }

                    else
                    {
                        dd[k][i]=dd[j][k];
                    }
                    dd[j][k]=0;
                    dd[k][j]=0;
                    dd[i][k]=0;
            }
        }
    }
}
```

LANGKAH PENGERJAAN

14. Mencari nilai centroid

```
float att[25][25];
int m,one,two;
one=1;
two=2;
m=2;
att[1][1]=matriksnew[1][1];
att[1][2]=matriksnew[1][2];
printf("\n");
for(i=1; i<=looping-1; i++)
{
    //for(j=1; j<=2; j++)
    //{
        for(k=2; k<=looping; k++)
        {
            //for(l=1; l<=2; l++)
            //{
                if(matriksnew[i][1]==matriksnew[k][one]||
                   matriksnew[i][1]==matriksnew[k][two]||
                   matriksnew[i][2]==matriksnew[k][one]||
                   matriksnew[i][2]==matriksnew[k][two])
                {
                    if(one==1)
                    {
                        m++;
                        att[i][m++]=matriksnew[k][2];
                    }
                    else if(two==2)
                    {
                        m++;
                        att[i][m++]=matriksnew[k][1];
                    }
                }
                else
                {
                    att[i][1]=matriksnew[i][1];
                    att[i][2]=matriksnew[i][2];
                }
            //}
        }
    //}
}
```

HASIL

```
att[1][1] = 20.000000
att[1][2] = 18.000000
att[1][3] = 20.000000
att[2][1] = 9.000000
att[2][2] = 7.000000
att[3][1] = 10.000000
att[3][2] = 8.000000
att[4][1] = 5.000000
att[4][2] = 9.000000
```

ANALISA

1. Metode Agglomerative Hierarchical Clustering (AHC) dengan pendekatan jarak single linkage, complete linkage dan average linkage dapat digunakan untuk membangun hirarki dari data dan mengelompokkannya.
2. Pada praktikum kali ini, optimasi Clustering dilakukan menggunakan metode Hierarchical Clustering dengan pencarian jarak terjauh (Complete Linkage) dimana pada metode ini didapat performansi paling baik daripada dua metode lainnya yaitu Single & Average Linkage
3. Proses pencarian centroid pada Optimasi Clustering dilakukan dengan 2 proses utama yaitu pencarian menggunakan metode K-Means Clustering, kemudian didapatkan data centroid yang nantinya akan digunakan dalam proses Hierarchical Clustering, kemudian hasil dari Hierarchical Clustering akan didapat data Centroid yang baik.