LAPORAN TUGAS PROGRAM

MACHINE LEARNING

CLUSTERING: K-MEANS

Nama : I Putu Indra Aristya

NIM : 1301154219

Kelas : IF39-09

Mata Kuliah : Pembelajaran Mesin

Universitas Telkom

BANDUNG

2018

I. Deskripsi Masalah

Dalam tugas program ini, mahasiswa diminta untuk melakukan klasterisasi terhadap data (*clustering*) dari data yang diberikan. Metode *clustering* yang digunakan adalah algoritma *K-Means*. Data yang diberikan adalah data latih dan data uji. Dari data latih yang diberikan, akan dicari nilai *k* terbaik untuk diterapkan ke dalam data uji.

II. Landasan Teori

Klasteriasi atau dalam bahasa inggris disebut *clustering* adalah salah satu proses membagi atau melakukan partisi kumpulan data atau objek ke dalam kelompok atau himpunan bagian yang dinamakan *cluster*. Klasterisasi adalah salah satu kelompok *unsupervised learning* dimana data yang digunakan tidak memiliki label sehingga dilakukan pengelompokan berdasar persamaan sifat. Salah satu algoritma klasterisasi yang dikenal dan digunakan dalam tugas program ini adalah algoritma *K-Means*.

K-Means adalah salah satu algoritma klasterisasi data dengan menggunakan konsep *centroid based*. Dimana pada *k-means* ini, ditentukan terlebih dahulu jumlah *cluster* dan ditentukan secara acak *centroid* sebagai titik pusat setiap *cluster*. Data yang memiliki jarak paling dekat dengan salah satu *centroid* akan masuk ke dalam satu *cluster* dengan titik pusat adalah *centroid* tersebut.

Algoritma *K-Means*.

1. Inisialisasi jumlah *cluster* (k) dan titik awal sebagai *centroid* sebanyak jumlah kluster
2. Hitung jarak setiap data ke setiap *centroid* dan tentukan data tersebut termasuk *cluster* yang mana
3. Tentukan *centroid* baru dengan menghitung nilai rata-rata dari data yang termasuk ke dalam setiap *cluster*
4. Lakukan langkah 2 – 3 hingga tidak terjadi perubahan nilai *centroid*
5. Hitung SSE

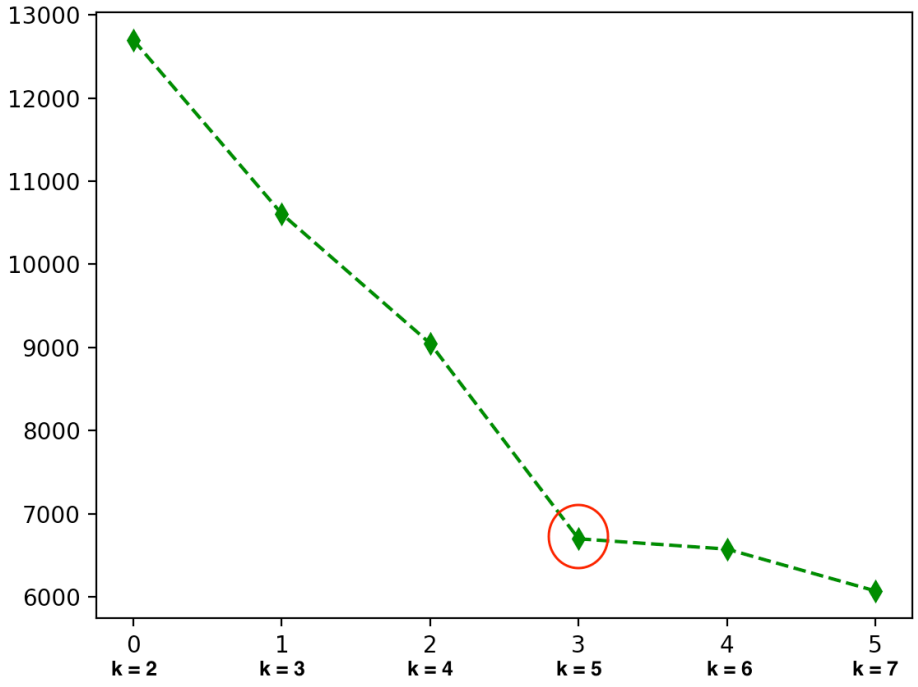
Dalam tugas ini, penghitungan jarak antara data dengan *centroid* menggunakan *ecluidean distance* dengan rumus sebagai berikut.

$$dist(A,B) = \sqrt{A - B^2}$$

Untuk penghitungan SSE yang dilakukan, adalah menjumlahkan jarak total setiap data terhadap titik *cluster*nya (*centroid*), digunakan rumus sebagai berikut.

$$SSE = \sum_{k=1}^{m} \sum_{i=0}^{n} dist(A,B)_i$$

Nilai *k* atau jumlah *cluster* pada tugas ini, saya tentukan dengan menggunakan metode *Elbow*. Dimana metode ini akan melihat nilai yang pertama saat akan menurun secara halus berdasarkan grafik nilai SSE pada Gambar 1.



Gambar 1 Grafik SSE

III. *Code* Program

Dalam tugas program ini, saya melakukan observasi terhadap nilai *k* atau jumlah *cluster* yang digunakan dan saya menentukan k = 5 dengan SSE pada eksekusi program saat ini adalah 1105.0635.

Pada kode program ini, saya melakukan random *centroid* awal dengan 2 cara, yaitu melakukan random dari nilai 0 – 36 atau melakukan random dari data yang sudah ada. Pada hasil yang didapatkan, dengan melakukan random dari nilai 0 – 36 terkadang akan mendapat nilai random yang kurang baik. Akibatnya, bisa saja salah satu *centroid* dianggap paling jauh dan tidak dekat sama sekali dengan data sehingga tidak akan ada data yang berada pada *cluster* tersebut. Hal ini mengakibatkan *error* pada program karena saat mencari nilai rata-rata untuk menentukan *centroid* yang baru tidak akan berhasil (karena pembagi = 0). Maka dari itu, diharapkan untuk melakukan *re-run* jika *error* tersebut terjadi.

```
Cent X:  [13.960000000000001, 18.710000000000001, 19.579999999999998, 24.059999999999999]
Cent Y:  [26.629999999999999, 5.8499999999999996, 11.33, 21.489999999999998]
SSE:  1105.0635
```

Gambar 2 Hasil *running* program cluster_TESTING.py

Berikut ditampilkan kode program untuk data uji.

```
#I Putu Indra Aristya - 1301154219

from numpy import genfromtxt, math
import numpy as np
import matplotlib.pyplot as plot
import random
from mpl_toolkits.mplot3d import Axes3D
fig = plot.figure()
ax = fig.add_subplot(111, projection='3d')

def ecluiDistance(a,b,x,y):
    return math.sqrt((a-x)**2) + ((b-y)**2)

dataTrain = genfromtxt('testing.csv',delimiter=',') #testing.csv adalah file data test

att1 = dataTrain[:,0]
att2 = dataTrain[:,1]

#------Code untuk visualisasi data latih dengan scatter plot------
# ax.scatter(xs=att1, ys=att2) # plot 3D
# plot.plot(att1,att2,'bo') #plot 2D
# plot.show()

hasil = dataTrain*0

# Code untuk melakukan random nilai sebagai centroid awal dari rentang 0 - 36 dan
dibulatkan 2 angka di belakang koma
xCent1 = (round(random.uniform(0,36),2))
yCent1 = (round(random.uniform(0,36),2))

xCent2 = (round(random.uniform(0,36),2))
yCent2 = (round(random.uniform(0,36),2))

xCent3 = (round(random.uniform(0,36),2))
yCent3 = (round(random.uniform(0,36),2))
```

```
xCent4 = (round(random.uniform(0,36),2))
yCent4 = (round(random.uniform(0,36),2))

xCent5 = (round(random.uniform(0,36),2))
yCent5 = (round(random.uniform(0,36),2))


# Code untuk melakukan random nilai sebagai centroid awal dari salah satu koordinat
data latih
# xCent1 = random.choice(att1)
# yCent1 = random.choice(att2)
#
# xCent2 = random.choice(att1)
# yCent2 = random.choice(att2)
#
# xCent3 = random.choice(att1)
# yCent3 = random.choice(att2)
#
# xCent4 = random.choice(att1)
# yCent4 = random.choice(att2)
#
# xCent5 = random.choice(att1)
# yCent5 = random.choice(att2)

sumxC1 = 0
sumyC1 = 0
sumxC2 = 0
sumyC2 = 0
sumxC3 = 0
sumyC3 = 0
sumxC4 = 0
sumyC4 = 0
sumxC5 = 0
sumyC5 = 0

count1 = 0
count2 = 0
count3 = 0
count4 = 0
count5 = 0

xCent = [xCent1,xCent2,xCent3,xCent4,xCent5]
yCent = [yCent1,yCent2,yCent3,yCent4,yCent5]
Cent = [xCent,yCent]
newCent = []
l = 1

while (Cent != newCent):
    sse = 0
    for i in range(0,100):
        # menghitung jarak setiap data ke setiap centroid
        clust1 = ecluiDistance(xCent1,yCent1,dataTrain[i,0],dataTrain[i,1])
        clust2 = ecluiDistance(xCent2,yCent2,dataTrain[i,0],dataTrain[i,1])
        clust3 = ecluiDistance(xCent3,yCent3,dataTrain[i,0],dataTrain[i,1])
        clust4 = ecluiDistance(xCent4,yCent4,dataTrain[i,0],dataTrain[i,1])
        clust5 = ecluiDistance(xCent5,yCent5,dataTrain[i,0],dataTrain[i,1])

        # menentukan data lebih dekat dengan centroid 1, 2, 3, 4 atau 5

        if (clust1 <= clust2):
            if (clust1 <= clust3):
                if (clust1 <= clust4):
                    if (clust1 <= clust5):
                        hasil[i,0] = clust1
                        hasil[i,1] = 1
        if (clust2 <= clust1):
            if (clust2 <= clust3):
                if (clust2 <= clust4):
                    if (clust2 <= clust5):
                        hasil[i, 0] = clust2
                        hasil[i, 1] = 2
        if (clust3 <= clust1):
            if (clust3 <= clust2):
                if (clust3 <= clust4):
```

```
                    if (clust3 <= clust5):
                        hasil[i, 0] = clust3
                        hasil[i, 1] = 3
        if (clust4 <= clust1):
            if (clust4 <= clust2):
                if (clust4 <= clust3):
                    if (clust4 <= clust5):
                        hasil[i, 0] = clust4
                        hasil[i, 1] = 4
        if (clust5 <= clust1):
            if (clust5 <= clust2):
                if (clust5 <= clust3):
                    if (clust5 <= clust4):
                        hasil[i, 0] = clust5
                        hasil[i, 1] = 5

    for i in range(0, 100):
        # menjumlahkan koordinat data di setiap cluster untuk mempermudah pencarian
centroid baru
        if hasil[i,1] == 1:
            sumxC1 = sumxC1 + dataTrain[i,0]
            sumyC1 = sumyC1 + dataTrain[i,1]
            count1 = count1 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 2:
            sumxC2 = sumxC2 + dataTrain[i,0]
            sumyC2 = sumyC2 + dataTrain[i,1]
            count2 = count2 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 3:
            sumxC3 = sumxC3 + dataTrain[i,0]
            sumyC3 = sumyC3 + dataTrain[i,1]
            count3 = count3 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 4:
            sumxC4 = sumxC4 + dataTrain[i,0]
            sumyC4 = sumyC4 + dataTrain[i,1]
            count4 = count4 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 5:
            sumxC5 = sumxC5 + dataTrain[i,0]
            sumyC5 = sumyC5 + dataTrain[i,1]
            count5 = count5 + 1
            sse = sse + hasil[i,0]

    xCent = [xCent1, xCent2, xCent3, xCent4, xCent5]
    yCent = [yCent1, yCent2, yCent3, yCent4, yCent5]
    Cent = [xCent, yCent]

    # menentukan centroid baru dengan mencari rata-rata dari koordinat data pada
setiap cluster
    xCent1 = round(sumxC1/count1,2)
    yCent1 = round(sumyC1/count1,2)

    xCent2 = round(sumxC2/count2,2)
    yCent2 = round(sumyC2/count2,2)

    xCent3 = round(sumxC3/count3,2)
    yCent3 = round(sumyC3/count3,2)

    xCent4 = round(sumxC4/count4,2)
    yCent4 = round(sumyC4/count4,2)

    xCent5 = round(sumxC5/count5,2)
    yCent5 = round(sumyC5/count5,2)

    xCent = [xCent1, xCent2, xCent3, xCent4, xCent5]
    yCent = [yCent1, yCent2, yCent3, yCent4, yCent5]
    newCent = [xCent,yCent]

    print("NewCent- ",l,": ",newCent)
    l = l+1

# xCent = [xCent1,xCent2,xCent3]
```

```
# yCent = [yCent1,yCent2,yCent3]
print("Cent X: ",xCent)
print("Cent Y: ",yCent)
print("SSE: ",sse)

s=121
ax.scatter(xs=att1,ys=att2,c=hasil[:,1])
ax.scatter(xs=xCent, ys=yCent,s=2*s,c='red',marker='^',alpha=.8)
plot.show()
```

Kode program untuk melakukan *training* dengan k = 5.

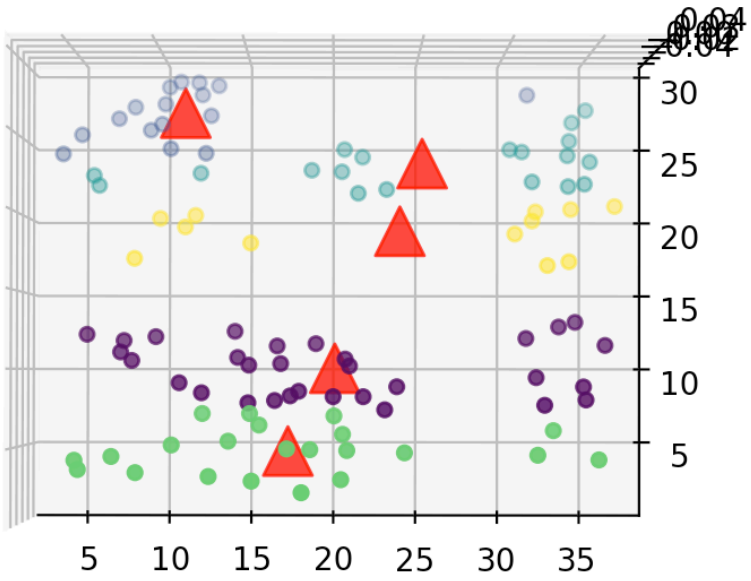
```
#I Putu Indra Aristya - 1301154219

from numpy import genfromtxt, math
import numpy as np
import matplotlib.pyplot as plot
import random
from mpl_toolkits.mplot3d import Axes3D
fig = plot.figure()
ax = fig.add_subplot(111, projection='3d')

def ecluiDistance(a,b,x,y):
    return math.sqrt((a-x)**2) + ((b-y)**2)

dataTrain = genfromtxt('dataset.csv',delimiter=',') #dataset.csv adalah file data latih

att1 = dataTrain[:,0]
att2 = dataTrain[:,1]

#------Code untuk visualisasi data latih dengan scatter plot------
# ax.scatter(xs=att1, ys=att2) # plot 3D
# plot.plot(att1,att2,'bo') #plot 2D
# plot.show()

hasil = dataTrain*0

# Code untuk melakukan random nilai sebagai centroid awal dari rentang 0 - 36 dan
dibulatkan 2 angka di belakang koma
xCent1 = (round(random.uniform(0,36),2))
yCent1 = (round(random.uniform(0,36),2))

xCent2 = (round(random.uniform(0,36),2))
yCent2 = (round(random.uniform(0,36),2))

xCent3 = (round(random.uniform(0,36),2))
yCent3 = (round(random.uniform(0,36),2))

xCent4 = (round(random.uniform(0,36),2))
yCent4 = (round(random.uniform(0,36),2))

xCent5 = (round(random.uniform(0,36),2))
yCent5 = (round(random.uniform(0,36),2))


# Code untuk melakukan random nilai sebagai centroid awal dari salah satu koordinat data
latih
# xCent1 = random.choice(att1)
# yCent1 = random.choice(att2)
#
# xCent2 = random.choice(att1)
# yCent2 = random.choice(att2)
#
# xCent3 = random.choice(att1)
# yCent3 = random.choice(att2)
#
# xCent4 = random.choice(att1)
# yCent4 = random.choice(att2)
#
# xCent5 = random.choice(att1)
```

```
# yCent5 = random.choice(att2)

sumxC1 = 0
sumyC1 = 0
sumxC2 = 0
sumyC2 = 0
sumxC3 = 0
sumyC3 = 0
sumxC4 = 0
sumyC4 = 0
sumxC5 = 0
sumyC5 = 0

count1 = 0
count2 = 0
count3 = 0
count4 = 0
count5 = 0

xCent = [xCent1,xCent2,xCent3,xCent4,xCent5]
yCent = [yCent1,yCent2,yCent3,yCent4,yCent5]
Cent = [xCent,yCent]
newCent = []
l = 1

while (Cent != newCent):
    sse = 0
    for i in range(0,688):
        # menghitung jarak setiap data ke setiap centroid
        clust1 = ecluiDistance(xCent1,yCent1,dataTrain[i,0],dataTrain[i,1])
        clust2 = ecluiDistance(xCent2,yCent2,dataTrain[i,0],dataTrain[i,1])
        clust3 = ecluiDistance(xCent3,yCent3,dataTrain[i,0],dataTrain[i,1])
        clust4 = ecluiDistance(xCent4,yCent4,dataTrain[i,0],dataTrain[i,1])
        clust5 = ecluiDistance(xCent5,yCent5,dataTrain[i,0],dataTrain[i,1])

        # menentukan data lebih dekat dengan centroid 1, 2, 3, 4 atau 5

        if (clust1 <= clust2):
            if (clust1 <= clust3):
                if (clust1 <= clust4):
                    if (clust1 <= clust5):
                        hasil[i,0] = clust1
                        hasil[i,1] = 1
        if (clust2 <= clust1):
            if (clust2 <= clust3):
                if (clust2 <= clust4):
                    if (clust2 <= clust5):
                        hasil[i, 0] = clust2
                        hasil[i, 1] = 2
        if (clust3 <= clust1):
            if (clust3 <= clust2):
                if (clust3 <= clust4):
                    if (clust3 <= clust5):
                        hasil[i, 0] = clust3
                        hasil[i, 1] = 3
        if (clust4 <= clust1):
            if (clust4 <= clust2):
                if (clust4 <= clust3):
                    if (clust4 <= clust5):
                        hasil[i, 0] = clust4
                        hasil[i, 1] = 4
        if (clust5 <= clust1):
            if (clust5 <= clust2):
                if (clust5 <= clust3):
                    if (clust5 <= clust4):
                        hasil[i, 0] = clust5
                        hasil[i, 1] = 5

    for i in range(0, 688):
        # menjumlahkan koordinat data di setiap cluster untuk mempermudah pencarian
centroid baru
        if hasil[i,1] == 1:
            sumxC1 = sumxC1 + dataTrain[i,0]
            sumyC1 = sumyC1 + dataTrain[i,1]
```

```
            count1 = count1 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 2:
            sumxC2 = sumxC2 + dataTrain[i,0]
            sumyC2 = sumyC2 + dataTrain[i,1]
            count2 = count2 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 3:
            sumxC3 = sumxC3 + dataTrain[i,0]
            sumyC3 = sumyC3 + dataTrain[i,1]
            count3 = count3 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 4:
            sumxC4 = sumxC4 + dataTrain[i,0]
            sumyC4 = sumyC4 + dataTrain[i,1]
            count4 = count4 + 1
            sse = sse + hasil[i,0]
        elif hasil[i,1] == 5:
            sumxC5 = sumxC5 + dataTrain[i,0]
            sumyC5 = sumyC5 + dataTrain[i,1]
            count5 = count5 + 1
            sse = sse + hasil[i,0]

    xCent = [xCent1, xCent2, xCent3, xCent4, xCent5]
    yCent = [yCent1, yCent2, yCent3, yCent4, yCent5]
    Cent = [xCent, yCent]

    # menentukan centroid baru dengan mencari rata-rata dari koordinat data pada setiap
cluster
    xCent1 = round(sumxC1/count1,2)
    yCent1 = round(sumyC1/count1,2)

    xCent2 = round(sumxC2/count2,2)
    yCent2 = round(sumyC2/count2,2)

    xCent3 = round(sumxC3/count3,2)
    yCent3 = round(sumyC3/count3,2)

    xCent4 = round(sumxC4/count4,2)
    yCent4 = round(sumyC4/count4,2)

    xCent5 = round(sumxC5/count5,2)
    yCent5 = round(sumyC5/count5,2)

    xCent = [xCent1, xCent2, xCent3, xCent4, xCent5]
    yCent = [yCent1, yCent2, yCent3, yCent4, yCent5]
    newCent = [xCent,yCent]

    print("NewCent- ",l,": ",newCent)
    l = l+1

# xCent = [xCent1,xCent2,xCent3]
# yCent = [yCent1,yCent2,yCent3]
print("Cent X: ",xCent)
print("Cent Y: ",yCent)
print("SSE: ",sse)

s=121
ax.scatter(xs=att1,ys=att2,c=hasil[:,1])
ax.scatter(xs=xCent, ys=yCent,s=2*s,c='red',marker='^',alpha=.8)
plot.show()
```

IV. Hasil Keluaran Program

Berikut adalah hasil klasterisasi data uji dengan k = 5.



Gambar 3 Hasil visualisasi data uji dengan k = 5.

V. Kesimpulan

Dengan menggunakan *k-means*, hasil yang didapatkan setiap melakukan eksekusi program (*running*) akan didapatkan hasil yang selalu berbeda. Hal ini dikarenakan *centroid* awal ditentukan dengan melakukan random yang artinya akan berada pada titik yang berbeda setiap melakukan *running* sehingga area *cluster* yang didapatkan akan bisa jadi berbeda, berubah-ubah dan tidak sesuai ekspektasi. Hasil *random centroid* tersebut juga menyebabkan *centroid* bisa jadi tidak dekat dengan setiap data, sehingga tidak dianggap sebagai *cluster* dan tidak bisa ditentukan titik baru *centroid* tersebut dan jumlah area *cluster* tidak sesuai yang diinisialisasi.

Daftar Pustaka

Suyanto. (2014). *Artificial Intelligence* (Vol. 2). Bandung: Informatika Bandung.