Nama : Mela Mai Anggraini  
Kelas : IF-40-09  
NIM : 1301160307

Tugas 2

1. *K-Means Clustering*

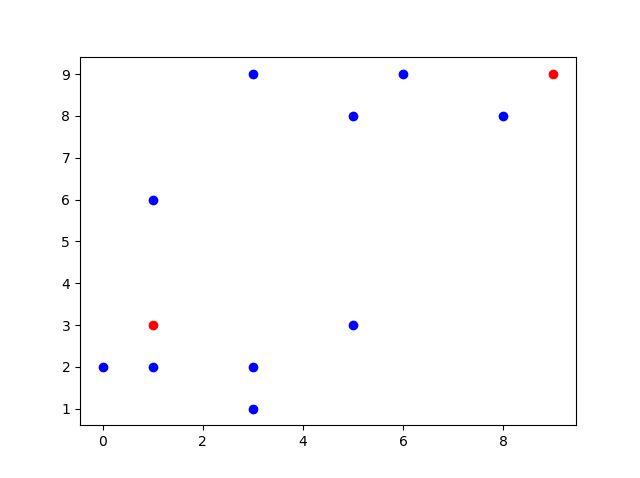
*K-Means Clustering* merupakan Metode pengelompokan partisi sederhana yang termasuk kedalam Unsupervised Learning. Tujuan dari metode ini yaitu mencari pola data yang tidak berlabel.

* Kelebihan
  + Waktu untuk pembelajaran relatif lebih cepat
  + Sangat umum penggunaannya
  + Menggunakan prinsip yang sederhana dapat dijelaskan dalam non-statistik
  + Sangat fleksibel.
* Kekurangan
  + Tidak menjamin menemukan kumpulan cluster yang optimal dikarenakan jumlah k cluster yang di set random.
  + Dapat terjadinya curse of dimensionality, apabila jarak antara cluster yang satu dengan yang lain memiliki banyak dimensi
  + Sensitive initial assign centroids.
* Contoh

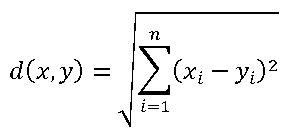
Diketahui:

|  |  |  |
| --- | --- | --- |
| Node | X | Y |
| P1 | 0 | 2 |
| P2 | 5 | 8 |
| P3 | 5 | 3 |
| P4 | 1 | 6 |
| P5 | 1 | 2 |
| P6 | 3 | 1 |
| P7 | 3 | 2 |
| P8 | 6 | 9 |
| P9 | 3 | 9 |
| P10 | 8 | 8 |

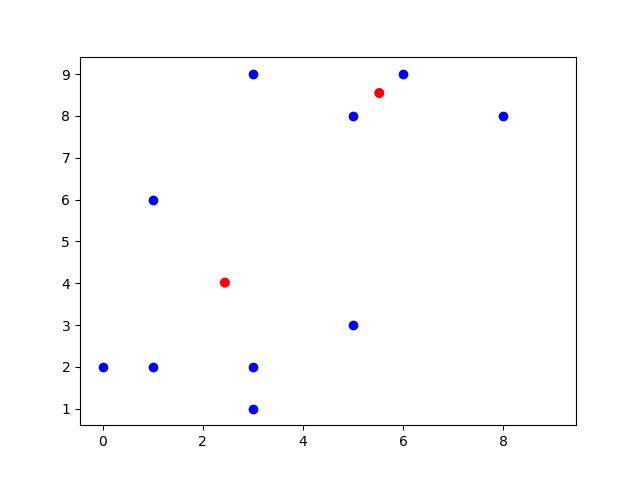
* 1. Inisiasi sejumlah K-cluster, pada contoh ini digunakan k=2.
  2. Inisiasi value random K (Centroids), pada contoh ini C1(9,9) dan C2(1,3). Berikut tampilannya.



* 1. Kemudian untuk setiap objek hitung jarak terdekat dengan setiap Centroids. Digunakan rumus Euclidean Distance.



* 1. Setelah itu, hitung rata-rata dari setiap cluster.
  2. Lakukan perulangan step b-c hingga centroid stabil/tidak berubah.
  3. Berikut hasil ketika proses selesai.



1. *Agglomerative Hierarchical Clustering*

*Agglomerative Hierarchical Clustering* merupakan proses pengelompokkan dengan 2 atau lebih objek yang mempunyai kesamaan paling dekat, kemudian proses diteruskan ke objek lainnya yang terdekat, sehingga menjadi kesatuan cluster secara hirarki. *Agglomerative Hierarchical Clustering* memiliki beberapa metode seperti, Single Linkage, Complete Linkage, Average Linkage, dll.

* Kelebihan
  + Mempercepat pengolahan data dan menghemat waktu karena data akan membentuk hierarki sehingga mempermudah dalam pembelajaran.
* Kekurangan
  + Seringnya terdapat kesalahan pada data outlier,
  + Perbedaan ukuran jarak yang digunakan,
  + Terdapat variable yang tidak relevan.
* Contoh

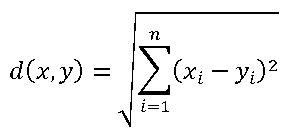
Diketahui :

|  |  |  |
| --- | --- | --- |
| Node | X | Y |
| P1 | 8 | 4 |
| P2 | 9 | 7 |
| P3 | 5 | 8 |
| P4 | 4 | 6 |
| P5 | 8 | 7 |
| P6 | 5 | 2 |
| P7 | 3 | 9 |
| P8 | 1 | 8 |

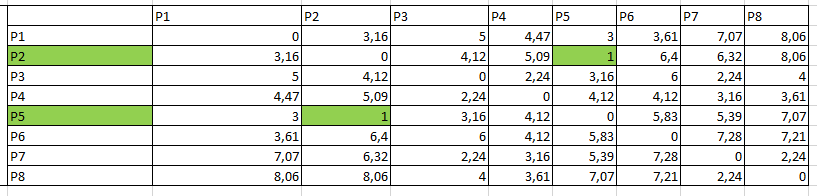
Jawab:

* 1. Hitung jarak setiap node terhadap node lainnya menggunakan rumus Euclidean dan petakan pada table.

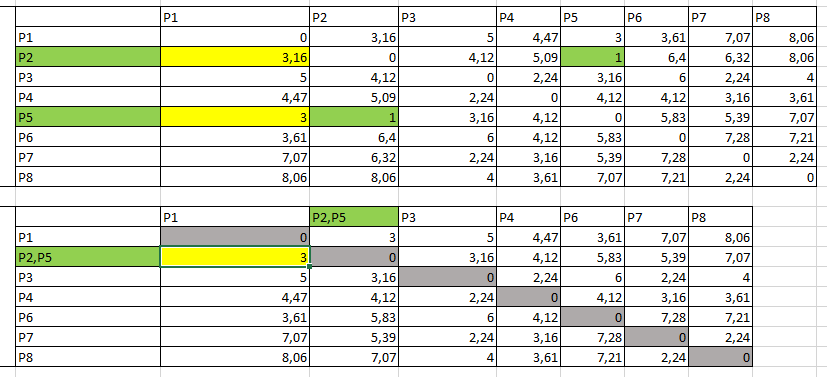
Rumus Euclidean Distance:



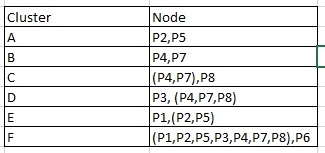
Hasil yang didapatkan:



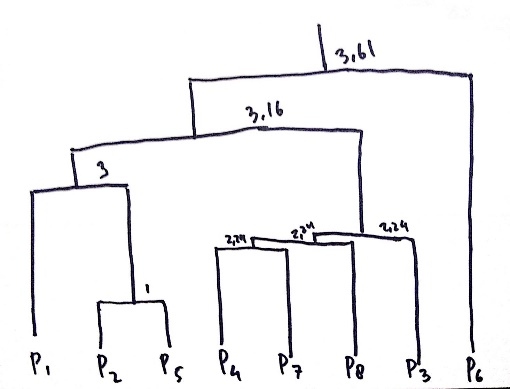
* 1. Dari data table diatas, Cari 2 nilai minimum(paling kecil) pada tiap baris node. Maka didapatkan node P2 dan P5. Node-node tersebut akan menjadi satu cluster.
  2. Setelah didapatkan 2 nilai minimum, maka gunakan salah satu metode Hierarchical Clstering seperti, Single Linkage, Complete Linkage, Average Linkage,dll. Pada contoh ini menggunakan metode Single Linkage. Single Linkage adalah metode mencari jarak node terkecil.



* 1. Lakukan perulangan dari step a - c hingga tersisa 1 cluster.
* Hasil Cluster dalam bentuk table



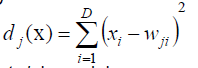
* Grafik Hierarchical Clustering.



1. *Self Organizing Map*

Analisis:

* Terdapat 600 objek yang setiap objeknya terdiri dari 2 atribut.
* Cara penyelesaian:
  1. Import file ‘Tugas 2 ML Genap 2018-2019 Dataset Tanpa Label.csv’.
  2. Inisiasi sigma, learning rate, tn(perubahan learning rate), dan t\_sigma(perubahan sigma). (lihat source code untuk lebih detail)
  3. Inisiasi Neuron yang berisi 2 atribut dengan nilai random antara 0 – 1.
  4. Normalisasi dataset tsb diantara range 0-1 sesuai dengan Neuron yang juga antara 0-1.
  5. Temukan/cari BMU(Best Matching Unit) dengan menghitung jarak/density antara setiap neuron terhadap data yang sedang di akses. Rumus density:



* 1. Setelah menemukan BMU, Hitung Topological Neighborhood of The BMU. Dengan rumus:



* 1. Selanjutnya sesuaikan Weight dari Neuron yang berdekatan dengan BMU. Pada step ini, di cari Delta Weight dari setiap neuron terhadap objek/data yang sedang di akses. Dengan rumus:



* 1. Setelah itu, update Weight Neuron. Dengan rumus:



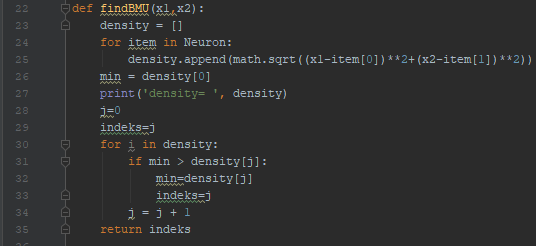
* 1. Lakukan perulangan dari step E hingga step H sebanyak jumlah objek.
  2. Setelah 1 iterasi selesai, update Sigma dan Learning rate. Dengan rumus:





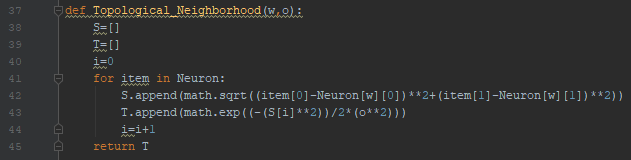
* 1. Lakukan perulangan sebanyak 3 kali(problem size= 3n)
* Fungsi yang digunakan:
  1. Find BMU

Fungsi ini bertujuan untuk mencari dan mengembalikan nilai baris/lokasi neuron BMU.



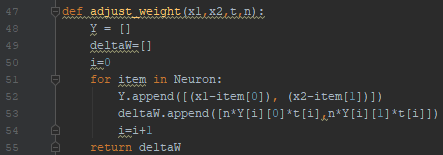
* 1. Topological Neighborhood

Fungsi ini bertujuan untuk menghitung jarak perpindahan neuron yang berdekatan



* 1. Adjust weight

Fungsi ini bertujuan untuk menghitung Delta Weight setiap neuron.



* 1. Update learning rate dan Sigma

