

Silahkan cari satu tutorial yang membahas tentang pengukuran jarak antar data. Buat ringkasan dari tutorial tersebut dan tulis dalam format ms word, lalu kumpulkan sebelum batas waktu. Sertakan sumber link tutorial yang diringkas.

## Tugas 1

Nama : I Gede Ery Santika  
NIM : 202420012  
Program : MTI

### Pengukuran Jarak Data Mining

#### 1. Euclidean Distance

Euclidean Distance yaitu mengukur dua buah objek berdasarkan jarak lurus atau garis lurus dalam Euclidean space.

Rumus pengukuran Euclidean Distance

$$d(x,y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

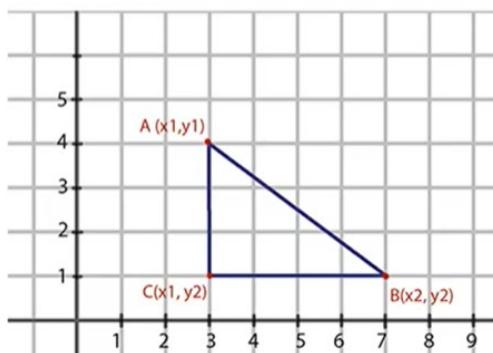
Dimana  $x$  dan  $y$  adalah dua objek data yang memiliki  $n$  atribut bernilai numerik

Contoh

- Diketahui dua buah titik (2,1) dan (3,2) hitunglah jarak menggunakan persamaan Euclidean

$$\begin{aligned} d(x,y) &= \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \\ &= \sqrt{(2-3)^2 + (1-2)^2} = \sqrt{2} \end{aligned}$$

- Diketahui tiga buah titik seperti gambar



Hitunglah jarak titik A ke B ,emggunakan persamaan Euclidean

Titik A (3,4), dan titik B (7,1)

$$\begin{aligned}
 d(x, y) &= \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \\
 &= \sqrt{(3 - 7)^2 + (4 - 1)^2} \\
 &= \sqrt{16 + 9} = \sqrt{25} = 5
 \end{aligned}$$

## 2. Manhattan Distance

Manhattan Distance yaitu mengukur jarak secara tegak lurus dari satu titik ke titik lain  
Rumus Manhattan Distance

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

Dimana  $x$  dan  $y$  adalah dua objek data yang memiliki  $n$  atribut bernilai numerik

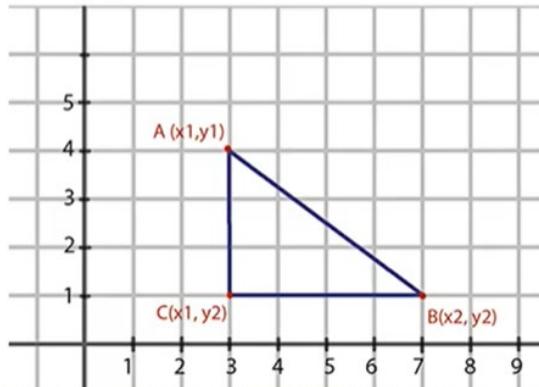
Contoh

- Diketahui dua buah titik (2,1) dan (3,2) hitunglah jarak menggunakan persamaan Menhattan Distance

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

$$= |2 - 3| + |1 - 2| = 2$$

- Diketahui tiga buah titik seperti gambar



Hitunglah jarak titik A ke B ,menggunakan persamaan Menhattan Distance

Titik A (3,4), dan titik B (7,1)

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

$$= |3 - 7| + |4 - 1| = 4 + 3 = 7$$

### 3. Cosine Similarity

Cosine Similarity yaitu mengukur kedekatan berdasarkan sudut Cosine. Ukuran jarak ini umumnya digunakan untuk data yang berupa vector dokumen yang memiliki ribuan atribut (kata) yang frekuensinya banyak yang bernilai 0.

Rumus Cosine Similarity

$$\text{Cos}(x, y) = \frac{x \cdot y}{\|x\| \|y\|}$$

Dimana  $x$  dan  $y$  adalah dua objek data yang memiliki  $n$  atribut bernilai numerik

Contoh

- Hitunglah nilai similaritas dengan metode *Cosine coefficient* antara kasus baru (P) dan pusat klaster (C) pada table

	Usia	G01	G02	G03	G04	G05
Kasus Baru ( P )	0.33	1	1	0	0	1
Pusat Klaster ( C )	0.59	0.65	0.17	0.4	0.9	0.6

$$\langle P, C \rangle = (P_1 * C_1) + (P_2 * C_2) + (P_3 * C_3) + (P_4 * C_4) + (P_5 * C_5) + (P_6 * C_6)$$

$$\langle P, C \rangle = (0.33 * 0.59) + (1 * 0.65) + (1 * 0.17) + (0 * 0.4) + (0 * 0.9) + (1 * 0.6) = 1.6$$

$$\|P\| = \sqrt{P_1^2 + P_2^2 + P_3^2 + P_4^2 + P_5^2 + P_6^2}$$

$$\|P\| = \sqrt{0.33^2 + 1^2 + 1^2 + 0^2 + 0^2 + 1^2} = 1.8$$

$$\|C\| = \sqrt{C_1^2 + C_2^2 + C_3^2 + C_4^2 + C_5^2 + C_6^2}$$

$$\|C\| = \sqrt{0.59^2 + 0.65^2 + 0.17^2 + 0.4^2 + 0.9^2 + 0.6^2} = 1.46$$

$$\text{Sim} = \text{COS}(P, C) = \frac{\langle P, C \rangle}{\|P\| \|C\|} = \frac{1.6}{1.8 * 1.46} = 0.63$$

---


$$\text{Dis Sim} = 1 - \text{Sim} = 1 - 0.63 = 0.37$$

Sumber : [https://youtu.be/QOJ8Fiy\\_W4Y](https://youtu.be/QOJ8Fiy_W4Y)

## **Mengukur Jarak Data**

### **Mengukur Jarak Tipe Numerik**

**Shirkhorshidi, A. S., Aghabozorgi, S., & Wah, T. Y. (2015). A comparison study on similarity and dissimilarity measures in clustering continuous data. PloS one, 10(12), e0144059.**

Salah satu tantangan dalam era ini dengan database yang memiliki banyak tipe data. Mengukur jarak adalah komponen utama dalam algoritma clustering berbasis jarak. Algoritma seperti Algoritma Partisioning misal K-Mean, K-medoid dan fuzzy c-mean dan rough clustering bergantung pada jarak untuk melakukan pengelompokan

Sebelum menjelaskan tentang beberapa macam ukuran jarak, kita mendefinisikan terlebih dahulu yaitu  $v_1, v_2$  vektor yang menyatakan  $v_1 = x_1, x_2, \dots, x_n, v_2 = y_1, y_2, \dots, y_n, v_1 = x_1, x_2, \dots, x_n, v_2 = y_1, y_2, \dots, y_n$ , dimana  $x_i, y_i$  disebut atribut. Ada beberapa ukuran similaritas atau ukuran jarak, diantaranya

#### ***Minkowski Distance***

Kelompok Minkowski diantaranya adalah Euclidean distance dan Manhattan distance, yang menjadi kasus khusus dari Minkowski distance. Minkowski distance dinyatakan dengan

$$d_{min} = (\sum_{i=1}^n |x_i - y_i|^m)^{1/m}, m \geq 1$$

dimana  $m$  adalah bilangan riil positif dan  $x_i, y_i$  adalah dua vektor dalam ruang dimensi  $n$ . Implementasi ukuran jarak Minkowski pada model clustering data atribut dilakukan normalisasi untuk menghindari dominasi dari atribut yang memiliki skala data besar.

#### ***Manhattan distance***

Manhattan distance adalah kasus khusus dari jarak Minkowski pada  $m = 1$ . Seperti Minkowski Distance, Manhattan distance sensitif terhadap outlier. Bila ukuran ini digunakan dalam algoritma clustering, bentuk cluster adalah hyper-rectangular. Ukuran ini didefinisikan dengan

$$d_{man} = \sum_{i=1}^n |x_i - y_i|$$

#### ***Euclidean distance***

Jarak yang paling terkenal yang digunakan untuk data numerik adalah jarak Euclidean. Ini adalah kasus khusus dari jarak Minkowski ketika  $m = 2$ . Jarak Euclidean berkinerja baik ketika digunakan untuk kumpulan data cluster kompak atau terisolasi. Meskipun jarak Euclidean sangat umum dalam pengelompokan, ia memiliki kelemahan: jika dua vektor data tidak memiliki nilai atribut yang sama, kemungkinan memiliki jarak yang lebih kecil daripada pasangan vektor data lainnya yang mengandung nilai atribut yang sama. Masalah lain dengan jarak Euclidean

sebagai fitur skala terbesar akan mendominasi yang lain. Normalisasi fitur kontinu adalah solusi untuk mengatasi kelemahan ini.

### **Average Distance**

Berkenaan dengan kekurangan dari Jarak Euclidian Distance diatas, rata rata jarak adalah versi modifikasi dari jarak Euclidian untuk memperbaiki hasil. Untuk dua titik  $x, y \in \mathbb{R}^n$ , rata-rata jarak didefinisikan dengan

$$d_{ave} = \frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2$$

### **Weighted euclidean distance**

Jika berdasarkan tingkatan penting dari masing masing atribut ditentukan, maka Weighted Euclidean distance adalah modifikasi lain dari jarak Euclidean distance yang dapat digunakan. Ukuran ini dirumuskan dengan

$$d_{we} = \sqrt{\sum_{i=1}^n w_i (x_i - y_i)^2}$$
 dimana  $w_i$  adalah bobot yang diberikan pada atribut ke  $i$ .

### **Chord distance**

Chord distance adalah salah satu ukuran jarak modifikasi Euclidean distance untuk mengatasi kekurangan dari Euclidean distance. Ini dapat dipecahkan juga dengan menggunakan skala pengukuran yang baik. Jarak ini dapat juga dihitung dari data yang tidak dinormalisasi. Chord distance didefinisikan dengan

$$d_{chord} = \sqrt{2 - 2 \sum_{i=1}^n x_i y_i / \|x\|_2 \|y\|_2}$$

dimana  $\|x\|_2$  adalah L<sub>2</sub>-norm  $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$

### **Mahalanobis distance**

Mahalanobis distance berdasarkan data berbeda dengan Euclidean dan Manhattan distances yang bebas antara data dengan data yang lain. Jarak Mahalanobis yang teratur dapat digunakan untuk mengekstraksi hyperellipsoidal clusters. Jarak Mahalanobis dapat mengurangi distorsi yang disebabkan oleh korelasi linier antara fitur dengan menerapkan transformasi pemutihan ke data atau dengan menggunakan kuadrat Jarak mahalanobis. Mahalanobis distance dinyatakan dengan

$$d_{mah} = \sqrt{(x-y) S^{-1} (x-y)^T}$$

dimana  $S$  adalah matrik covariance data.

### **Cosine measure**

Ukuran Cosine similarity lebih banyak digunakan dalam similaritas dokumen dan dinyatakan dengan

$$\text{Cosine}(x, y) = \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}}$$

dimana  $\|y\|_2$  adalah Euclidean norm dari vektor  $y=(y_1,y_2,\dots,y_n)$  didefinisikan dengan  $\|y\|_2=\sqrt{y_{11}+y_{22}+\dots+y_{nn}}$

### Pearson correlation

Pearson correlation banyak digunakan dalam data expresi gen. Ukuran similaritas ini menghitung similaritas antara dua bentuk pola expresi gen. Pearson correlation didefinisikan dengan

$$\text{Pearson}(x,y) = \frac{\sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^n (x_i - \mu_x)^2} \sqrt{\sum_{i=1}^n (y_i - \mu_y)^2}}$$

The Pearson correlation kelebihannya adalah sensitif terhadap outlier

### Mengukur Jarak Atribut Binary

Mari kita lihat similaritas dan dissimilarity untuk objek yang dijelaskan oleh atribut biner simetris atau asimetris. Atribut biner hanya memiliki dua status: 0 dan 1. Contoh atribut perokok menggambarkan seorang pasien, misalnya, 1 menunjukkan bahwa pasien merokok, sedangkan 0 menunjukkan pasien tidak merokok. Memperlakukan atribut biner sebagai atribut numerik tidak diperkenankan. Oleh karena itu, metode khusus untuk data biner diperlukan untuk membedakan komputasi.

Jadi, bagaimana kita bisa menghitung ketidaksamaan antara dua atribut biner? "Satu pendekatan melibatkan penghitungan matriks ketidaksamaan dari data biner yang diberikan. Jika semua atribut biner dianggap memiliki bobot yang sama, kita memiliki tabel kontingensi  $2 \times 2 \times 2$  di mana  $qq$  adalah jumlah atribut yang sama dengan 1 untuk kedua objek  $ii$  dan  $jj$ ,  $rr$  adalah jumlah atribut yang sama dengan 1 untuk objek  $ii$  tetapi 0 untuk objek  $jj$ ,  $ss$  adalah jumlah atribut yang sama dengan 0 untuk objek  $ii$  tetapi 1 untuk objek  $jj$ , dan  $tt$  adalah jumlah atribut yang sama dengan 0 untuk kedua objek  $ii$  dan  $jj$ . Jumlah total atribut adalah  $pp$ , di mana  $p=q+r+s+t$

Inginlah bahwa untuk atribut biner simetris, masing-masing nilai bobot yang sama. Dissimilarity yang didasarkan pada atribut asymmetric binary disebut symmetric binary dissimilarity. Jika objek  $i$  dan  $j$  dinyatakan sebagai atribut biner simetris, maka dissimilarity antara  $ii$  dan  $jj$  adalah

$$d(i,j) = r + s + t$$

Untuk atribut biner asimetris, kedua kondisi tersebut tidak sama pentingnya, seperti hasil positif (1) dan negatif (0) dari tes penyakit. Diberikan dua atribut biner asimetris, pencocokan keduanya 1 (kecocokan positif) kemudian dianggap lebih signifikan daripada kecocokan negatif. Ketidaksamaan berdasarkan atribut-atribut ini disebut asimetris biner dissimilarity, di mana jumlah kecocokan negatif,  $t$ , dianggap tidak penting dan dengan demikian diabaikan. Berikut perhitungannya

$$d(i,j) = r + sq + r + sd(i,j) = r + sq + r + s$$

Kita dapat mengukur perbedaan antara dua atribut biner berdasarkan pada disimilarity. Misalnya, biner asimetris kesamaan antara objek  $i$  dan  $j$  dapat dihitung dengan

$$\text{sim}(i,j) = \frac{qq + r}{qq + r + s} = 1 - d(i,j)$$

Persamaan similarity ini disebut dengan **Jaccard coefficient**

### Mengukur Jarak Tipe categorical

*Li, C., & Li, H. (2010). A Survey of Distance Metrics for Nominal Attributes. JSW, 5(11), 1262-1269.*

#### Overlay Metric

Ketika semua atribut adalah bertipe nominal, ukuran jarak yang paling sederhana adalah dengan Overlay Metric (OM) yang dinyatakan dengan

$$d(x,y) = \sum_{i=1}^n \delta(a_i(x), a_i(y))$$

dimana  $n$  adalah banyaknya atribut,  $a_i(x)$  dan  $a_i(y)$  adalah nilai atribut ke  $i$  yaitu  $A_i$  dari masing-masing objek  $x$  dan  $y$ ,  $\delta(a_i(x), a_i(y))$  adalah 0 jika  $a_i(x) = a_i(y)$  dan 1 jika sebaliknya.

OM banyak digunakan oleh instance-based learning dan locally weighted learning. Jelas sekali, ini sedikit beruk untuk mengukur jarak antara masing-masing pasangan sample, karena gagal memanfaatkan tambahan informasi yang diberikan oleh nilai atribut nominal yang bisa membantu dalam generalisasi.

#### Value Difference Metric (VDM)

VDM dikenalkan oleh Standfill and Waltz, versi sederhana dari VDM tanpa skema pembobotan didefinisikan dengan

$$d(x,y) = \sum_{i=1}^n \sum_{c=1}^C |P(c|a_i(x)) - P(c|a_i(y))|$$

dimana  $C$  adalah banyaknya kelas,  $P(c|a_i(x))$  adalah probabilitas bersyarat dimana kelas  $x$  adalah  $c$  dari atribut  $A_i$ , yang memiliki nilai  $a_i(x)$ ,  $P(c|a_i(y))$  adalah probabilitas bersyarat dimana kelas  $y$  adalah  $c$  dengan atribut  $A_i$  memiliki nilai  $a_i(y)$

VDM mengasumsikan bahwa dua nilai dari atribut adalah lebih dekat jika memiliki klasifikasi sama. Pendekatan lain berbasis probabilitas adalah SFM (Short and Fukunaga Metric) yang kemudian dikembangkan oleh Myles dan Hand dan didefinisikan dengan

$$d(x,y) = \sum_{c=1}^C |P(c|x) - P(c|y)|$$

dimana probabilitas keanggotaan kelas diestimasi dengan  $P(c|x)P(c|y)$  dan  $P(c|y)P(c|y)$  didekati dengan Naive Bayes,  
**Minimum Risk Metric (MRM)**

Ukuran ini dipresentasikan oleh Blanzieri and Ricci, berbeda dari SFM yaitu meminimumkan selisih antara kesalahan berhingga dan kesalahan asymptotic. MRM meminimumkan risk of misclassification yang didefinisikan dengan

$$d(x,y) = \sum_{c=1}^C P(c|x)(1-P(c|y))$$

#### Mengukur Jarak Tipe Ordinal

**Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques.**  
Elsevier.

Nilai-nilai atribut ordinal memiliki urutan atau peringkat, namun besarnya antara nilai-nilai berturut-turut tidak diketahui. Contohnya tingkatan kecil, sedang, besar untuk atribut ukuran. Atribut ordinal juga dapat diperoleh dari diskritisasi atribut numerik dengan membuat rentang nilai ke dalam sejumlah kategori tertentu. Kategori-kategori ini disusun dalam peringkat. Yaitu, rentang atribut numerik dapat dipetakan ke atribut ordinal  $f_f$  yang memiliki  $M_f$  status. Misalnya, kisaran suhu atribut skala-skala (dalam Celcius) dapat diatur ke dalam status berikut:  $-30$  hingga  $-10$ ,  $-10$  hingga  $10$ ,  $10$  hingga  $30$ , masing-masing mewakili kategori suhu dingin, suhu sedang, dan suhu hangat.  $M_f$  adalah jumlah keadaan yang dapat dilakukan oleh atribut ordinal memiliki. State ini menentukan peringkat  $1, \dots, M_f$ . Perlakuan untuk atribut ordinal adalah cukup sama dengan atribut numerik ketika menghitung dissimilarity antara objek. Misalkan  $f_f$  adalah atribut-atribut dari atribut ordinal dari  $n$  objek. Menghitung dissimilarity terhadap  $f$  fitur sebagai berikut:

- Nilai  $f_f$  untuk objek ke- $i$  adalah  $x_{if}$ , dan  $f_f$  memiliki  $M_f$  status urutan, mewakili peringkat  $1, \dots, M_f$ . Ganti setiap  $x_{if}$  dengan peringkatnya,  $r_{if} \in \{1, \dots, M_f\}$
- Karena setiap atribut ordinal dapat memiliki jumlah state yang berbeda, diperlukan untuk memetakan rentang setiap atribut ke  $[0,0, 1.0]$  sehingga setiap atribut memiliki bobot yang sama. Perlakukan normalisasi data dengan mengganti peringkat  $r_{if}$  dengan  $z_{if} = r_{if} - 1/M_f - 1$
- Dissimilarity kemudian dihitung dengan menggunakan ukuran jarak seperti atribut numerik dengan data yang baru setelah ditransformasi  $z_{if}$

#### Menghitung Jarak Tipe Campuran

**Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. Journal of artificial intelligence research, 6, 1-34.**

Menghitung ketidaksamaan antara objek dengan atribut campuran yang berupa nominal, biner simetris, biner asimetris, numerik, atau ordinal yang ada pada kebanyakan database dapat dinyatakan dengan memproses semua tipe atribut

secara bersamaan. Salah satu teknik tersebut menggabungkan atribut yang berbeda ke dalam matriks ketidaksamaan tunggal dan menyatakannya dengan skala interval antar  $[0,0,1,0][0,0,1,0]$ . Misalkan data berisi atribut pp tipe campuran.

Ketidaksamaan (disimilarity ) antara objek  $i$  dan  $j$  dinyatakan dengan

$$d(i,j) = \sum_{pf=1}^p \delta(f)ij d(f)ij \sum_{pf=1}^p \delta(f)ij d(i,j) = \sum_{f=1}^p \delta(f)ij(f) d(f)ij(f) \sum_{f=1}^p \delta(f)ij(f)$$

dimana  $\delta(f)ij=0$  jika  $x_{if}=x_{jf}$  atau  $x_{if}\neq x_{jf}$  adalah hilang (i.e., tidak ada pengukuran dari atribut  $f$  untuk objek  $i$  atau objek  $j$ )

- jika  $x_{if}=x_{jf}=0$   $x_{if}=x_{jf}=0$  dan
- atribut  $f$  adalah binary asymmetric,  
selain itu  $\delta(f)ij=1$

Kontribusi dari atribut  $f$  untuk dissimilarity antara  $i$  dan  $j$  (yaitu.  $d_{ij}d_{if}$ ) dihitung bergantung pada tipenya,

- Jika  $f$  adalah numerik,  $d_{ij} = \|x_{if}-x_{jf}\|/\max_h x_{hf} - \min_h x_{hf}$   $d_{if} = \|x_{if}-x_{jf}\|/\max_h x_{hf} - \min_h x_{hf}$ , di mana  $h$  menjalankan semua nilai objek yang tidak hilang untuk atribut  $f$
- Jika  $f$  adalah nominal atau binary,  $d_{ij} = 0$  jika  $x_{if}=x_{jf}$   $x_{if}\neq x_{jf}$ , sebaliknya  $d_{ij}=1$
- Jika  $f$  adalah ordinal maka hitung rangking  $r_{if}r_{if}$  dan  $Z_{if} = r_{if} - 1M_f - 1$   $Z_{if} = r_{if} - 1M_f - 1$ , dan perlakukan  $Z_{if}Z_{if}$  sebagai numerik.

## Algoritma Hierarchical Clustering

Hierarchical Clustering adalah metode analisis kelompok yang berusaha untuk membangun sebuah hierarki kelompok data.

Strategi pengelompokannya umumnya ada 2 jenis yaitu **Agglomerative (Bottom-Up)** dan **Devisive (Top-Down)**.

### Langkah Algoritma Agglomerative Hierarchical Clustering :

1. Hitung Matrik Jarak antar data.
2. Gabungkan dua kelompok terdekat berdasarkan parameter kedekatan yang ditentukan.
3. Perbarui Matrik Jarak antar data untuk merepresentasikan kedekatan diantara kelompok baru dan kelompok yang masih tersisa.
4. Ulangi langkah 2 dan 3 hingga hanya satu kelompok yang tersisa.

Membentuk Matrik Jarak, misal dengan **Manhattan Distance** :

$$D = \sum_{i=1}^n |b_i - a_i|$$

*Persamaan Manhattan Distance*

atau menggunakan **Euclidian Distance** :

$$D(a, b) = \sqrt{\sum_{i=1}^n (b_i - a_i)^2}$$

*Persamaan Euclidean Distance*

Beberapa metode Pengelompokan **Agglomerative Hierarchical** :

a. **Single Linkage (Jarak Terdekat)**

$$d_{uv} = \min_{single-linkage} \{d_{uv}\}, d_{uv} \in D$$

b. **Complete Linkage (Jarak Terjauh)**

$$d_{uv} = \max_{complete-linkage} \{d_{uv}\}, d_{uv} \in D$$

c. **Average Linkage (Jarak Rata-Rata)**

$$d_{uv} = average \{d_{uv}\}, d_{uv} \in D$$

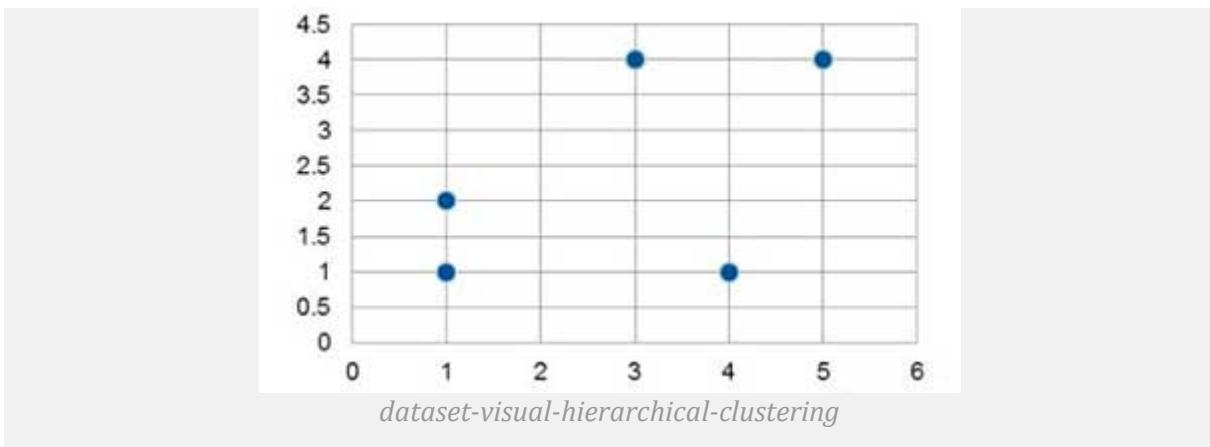
## 1. Contoh Soal Perhitungan

Perhatikan dataset berikut :

Data	Fitur x	Fitur y
1	1	1
2	4	1
3	1	2
4	3	4
5	5	4

*dataset-hierarchical-clustering*

Kelompokkan dataset tersebut dengan menggunakan metode AHC (Single Linkage, Complete Linkage dan Average Linkage) menggunakan jarak Manhattan !



Menghitung Jarak Pada Semua Pasangan dua data :

$$D = \sum_{i=1}^n |b_i - a_i|$$

**informaticalogi.com**

*Persamaan Manhattan Distance*

$$D_{man} (Data1, Data1) = |1-1| + |1-1| = 0$$

$$D_{man} (Data1, Data2) = |1-4| + |1-1| = 3$$

$$D_{man} (Data1, Data3) = |1-1| + |1-2| = 1$$

$$D_{man} (Data1, Data4) = |1-3| + |1-4| = 5$$

$$D_{man} (Data1, Data5) = |1-5| + |1-4| = 7$$

$$D_{man} (Data2, Data3) = |4-1| + |1-2| = 4$$

$$D_{man} (Data2, Data4) = |4-3| + |1-4| = 4$$

$$D_{man} (Data2, Data5) = |4-5| + |1-4| = 4$$

$$D_{man} (Data3, Data4) = |1-3| + |2-4| = 4$$

$$D_{man} (Data3, Data5) = |1-5| + |2-4| = 6$$

$$D_{man} (Data4, Data5) = |3-5| + |4-4| = 2$$

Dman	1	2	3	4	5
1	0	3	1	5	7
2	3	0	4	4	4
3	1	4	0	4	6
4	5	4	4	0	2
5	7	4	6	2	0

*Dman-hierarchical-clustering*

## 2. Metode Single Linkage

Dengan memperlakukan data sebagai kelompok, selanjutnya kita pilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{13}) = 1$$

Terpilih kelompok 1 dan 3, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok (1 dan 3) dengan kelompok lain yang tersisa, yaitu 2, 4 dan 5.

$$d_{(13)2} = \min \{d_{12}, d_{32}\} = \min \{3,4\} = 3$$

$$d_{(13)4} = \min \{d_{14}, d_{34}\} = \min \{5,4\} = 4$$

$$d_{(13)5} = \min \{d_{15}, d_{35}\} = \min \{7,6\} = 6$$

Dengan menghapus baris-baris dan kolom-kolom matrik jarak yang bersesuaian dengan kelompok 1 dan 3, serta menambahkan baris dan kolom untuk kelompok (13).

D <sub>man</sub>	(13)	2	4	5
(13)	0	3	4	6
2	3	0	4	4
4	4	4	0	2
5	6	4	2	0

D<sub>man</sub>-hierarchical-clustering-(2)

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{45}) = 2$$

Menghitung jarak antar kelompok (4 dan 5) dengan kelompok lain yang tersisa, yaitu (13) dan 2.

$$d_{(45)(13)} = \min \{d_{41}, d_{43}, d_{51}, d_{53}\} = \min \{5,4,7,6\} = 4$$

$$d_{(45)2} = \min \{d_{42}, d_{52}\} = \min \{4,4\} = 4$$

Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok 4 dan 5, serta menambahkan baris dan kolom untuk kelompok (45)

Dman	(45)	(13)	2
(45)	0	4	4
(13)	4	0	3
2	4	3	0

Dman-hierarchical-clustering-(3)

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{(13)2}) = 3$$

Terpilih kelompok (13) dan 2, sehingga kedua kelompok ini digabungkan.  
(Melanjutkan pengelompokan).

Menghitung jarak antar kelompok ((13) dan 2) dengan kelompok lain yang tersisa, yaitu (45).

$$d_{(132)(45)} = \min \{d_{14}, d_{15}, d_{34}, d_{35}, d_{24}, d_{25}\} = \min \{5, 7, 4, 6, 4, 4\} = 4$$

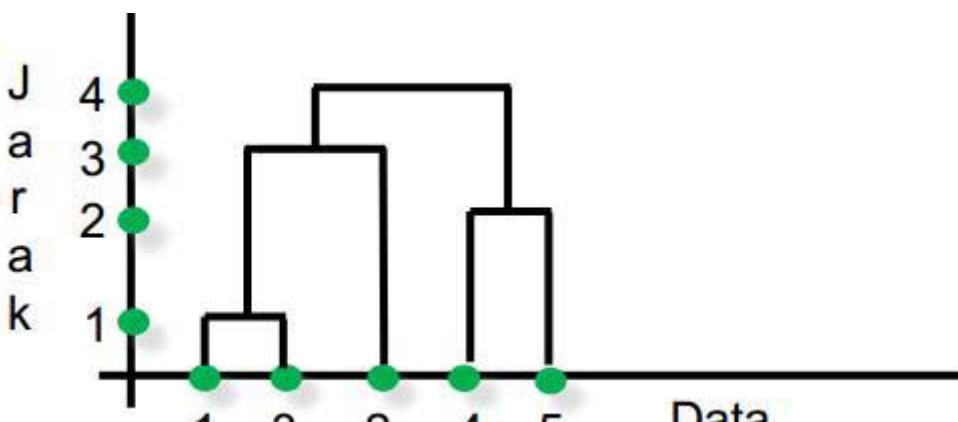
Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok (13) dan 2, serta menambahkan baris dan kolom untuk kelompok (123).

Dman	(132)	(45)
(132)	0	4
(45)	4	0

Dman-hierarchical-clustering-(4)

Jadi kelompok (132) dan (45) digabung untuk menjadi kelompok tunggal dari lima data, yaitu kelompok (13245) dengan jarak terdekat 4.

Berikut Dendogram Hasil Metode Single Linkage :



dendogram-single-linkage

### 3. Metode Complete Linkage

Dengan memperlakukan data sebagai kelompok, selanjutnya kita pilih jarak dua kelompok yang terkecil.

<b>Dman</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>1</b>	0	3	1	5	7
<b>2</b>	3	0	4	4	4
<b>3</b>	1	4	0	4	6
<b>4</b>	5	4	4	0	2
<b>5</b>	7	4	6	2	0

*Dman-hierarchical-clustering*

$$\min(D_{man}) = \min(d_{13}) = 1$$

Terpilih kelompok 1 dan 3, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok (1 dan 3) dengan kelompok lain yang tersisa, yaitu 2, 4 dan 5.

$$d_{(13)2} = \max \{d_{12}, d_{32}\} = \max \{3,4\} = 4$$

$$d_{(13)4} = \max \{d_{14}, d_{34}\} = \max \{5,4\} = 5$$

$$d_{(13)5} = \max \{d_{15}, d_{35}\} = \max \{7,6\} = 7$$

Dengan menghapus baris-baris dan kolom-kolom matrik jarak yang bersesuaian dengan kelompok 1 dan 3, serta menambahkan baris dan kolom untuk kelompok (13).

<b>Dman</b>	<b>(13)</b>	<b>2</b>	<b>4</b>	<b>5</b>
<b>(13)</b>	0	4	5	7
<b>2</b>	4	0	4	4
<b>4</b>	5	4	0	2
<b>5</b>	7	4	2	0

*Dman-hierarchical-clustering-complete*

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{45}) = 2$$

Dengan menghapus baris-baris dan kolom-kolom matrik jarak yang bersesuaian dengan kelompok 1 dan 3, serta menambahkan baris dan kolom untuk kelompok (13).

Menghitung jarak antar kelompok (4 dan 5) dengan kelompok lain yang tersisa, yaitu (13) dan 2.

$$d_{(45)(13)} = \max \{d_{41}, d_{43}, d_{51}, d_{53}\} = \max \{5,4,7,6\} = 7$$

$$d_{(45)2} = \max \{d_{42}, d_{52}\} = \max \{4, 4\} = 4$$

Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok 4 dan 5, serta menambahkan baris dan kolom untuk kelompok (45).

Dman	(45)	(13)	2
(45)	0	7	4
(13)	7	0	4
2	4	4	0

Dman-hierarchical-clustering-complete-(2)

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{(45)2}) = 4$$

Terpilih kelompok (45) dan 2, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok ((45) dan 2) dengan kelompok lain yang tersisa, yaitu (13).

$$d_{(452)(13)} = \min \{d_{41}, d_{43}, d_{51}, d_{53}, d_{21}, d_{23}\} = \max \{5, 4, 7, 6, 3, 4\} = 7$$

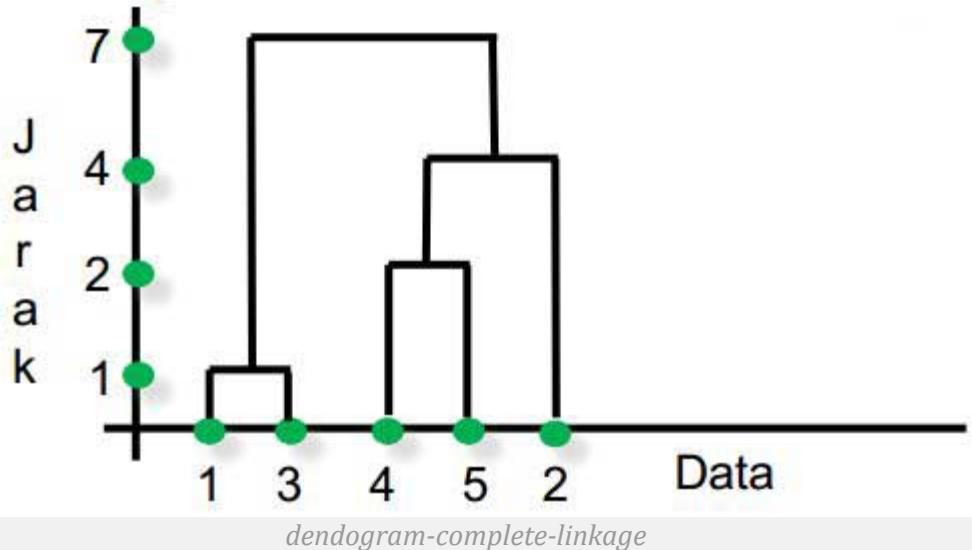
Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok (45) dan 2, serta menambahkan baris dan kolom untuk kelompok (452).

Dman	(452)	(13)
(452)	0	7
(13)	7	0

Dman-hierarchical-clustering-complete-(3)

Jadi kelompok (452) dan (13) digabung untuk menjadi kelompok tunggal dari lima data, yaitu kelompok (13452) dengan jarak terdekat 7.

Berikut Dendogram Hasil Metode Complete Linkage :



#### 4. Metode Average Linkage

Dengan memperlakukan data sebagai kelompok, selanjutnya kita pilih jarak dua kelompok yang terkecil.

<b>D<sub>man</sub></b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>1</b>	0	3	1	5	7
<b>2</b>	3	0	4	4	4
<b>3</b>	1	4	0	4	6
<b>4</b>	5	4	4	0	2
<b>5</b>	7	4	6	2	0

*D<sub>man</sub>-hierarchical-clustering*

$$\min(D_{\text{man}}) = \min(d_{13}) = 1$$

Terpilih kelompok 1 dan 3, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok (1 dan 3) dengan kelompok lain yang tersisa, yaitu 2, 4 dan 5.

$$d_{(13)2} = \text{average } \{d_{12}, d_{32}\} = \text{average } \{3, 4\} = (3+4) / 2 = 3.5$$

$$d_{(13)4} = \text{average } \{d_{14}, d_{34}\} = \text{average } \{5, 4\} = (5+4) / 2 = 4.5$$

$$d_{(13)5} = \text{average } \{d_{15}, d_{35}\} = \text{average } \{7, 6\} = (7+6) / 2 = 6.5$$

Dengan menghapus baris-baris dan kolom-kolom matrik jarak yang bersesuaian dengan kelompok 1 dan 3, serta menambahkan baris dan kolom untuk kelompok (13).

Dman	(13)	2	4	5
(13)	0	3.5	4.5	6.5
2	3.5	0	4	4
4	4.5	4	0	2
5	6.5	4	2	0

*Dman-hierarchical-clustering-average*

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{45}) = 2$$

Menghitung jarak antar kelompok (4 dan 5) dengan kelompok lain yang tersisa, yaitu (13) dan 2.

$$d_{(45)(13)} = \text{average } \{d_{41}, d_{43}, d_{51}, d_{53}\} = \text{average } \{5, 4, 7, 6\} = (5+4+7+6) / 4 = 5.25$$

$$d_{(45)2} = \text{average } \{d_{42}, d_{52}\} = \text{average } \{4, 4\} = (4+4) / 2 = 4$$

Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok 4 dan 5, serta menambahkan baris dan kolom untuk kelompok (45).

Dman	(45)	(13)	2
(45)	0	5.25	4
(13)	5.25	0	3.5
2	4	3.5	0

*Dman-hierarchical-clustering-average-(2)*

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{(13)2}) = 3.5$$

Terpilih kelompok (13) dan 2, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok ((13) dan 2) dengan kelompok lain yang tersisa, yaitu (45).

$$d_{(452)(13)} = \text{average } \{d_{14}, d_{15}, d_{34}, d_{35}, d_{24}, d_{25}\} = \text{average } \{5, 7, 7, 4, 6, 4\} = (5+7+7+4+6+4) / 6 = 5$$

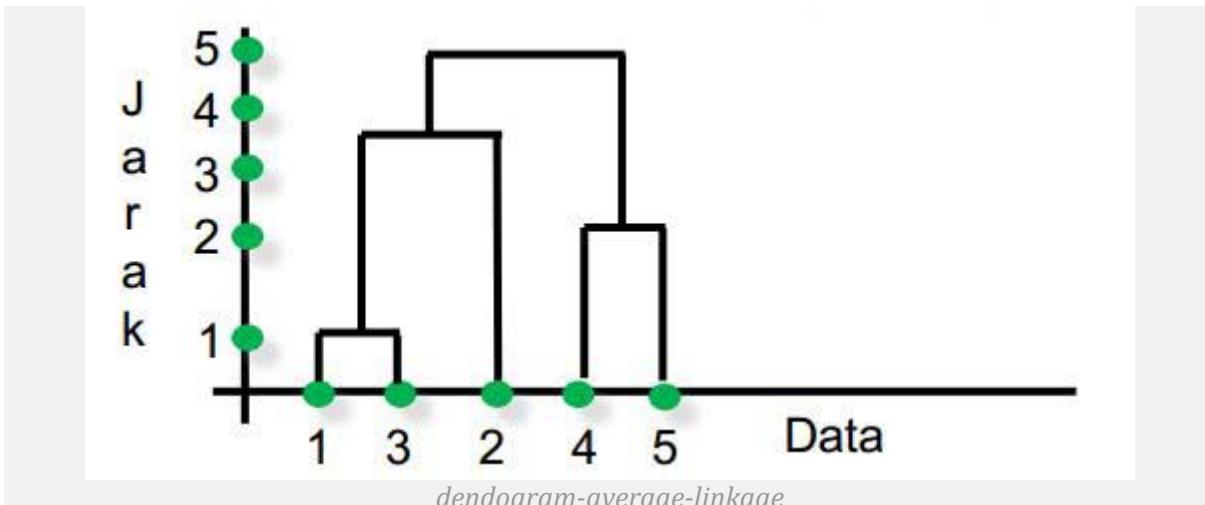
Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok (45) dan 2, serta menambahkan baris dan kolom untuk kelompok (452).

Dman	(132)	(45)
(132)	0	5
(45)	5	0

*Dman-hierarchical-clustering-average-(3)*

Jadi kelompok (132) dan (45) digabung untuk menjadi kelompok tunggal dari lima data, yaitu kelompok (13245) dengan jarak terdekat 5.

Berikut Dendogram Hasil Metode Average Linkage :



<https://informatikalogi.com/algoritma-hierarchical-clustering/>

<https://mulaab.github.io/datamining/memahami-data/>

## **Data Mining**

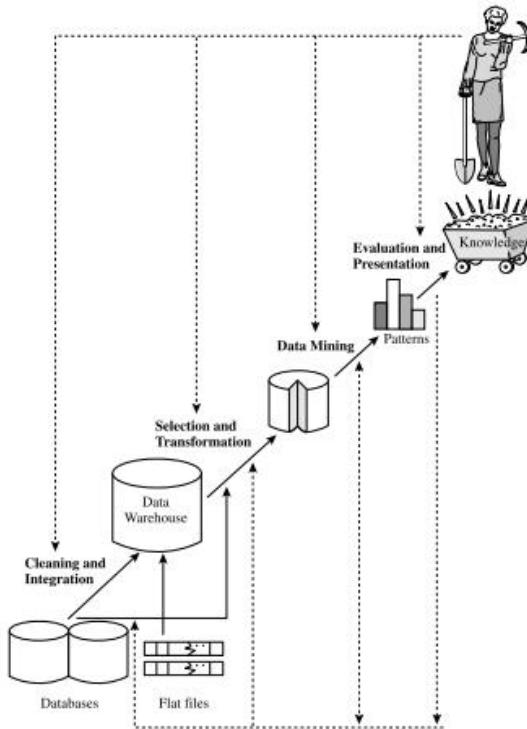
Secara sederhana, data mining dapat diartikan sebagai proses mengekstrak atau menggali knowledge yang ada pada sekumpulan data. Informasi dan knowledge yang didapat tersebut dapat digunakan pada banyak bidang, seperti manajemen bisnis, pendidikan, kesehatan dan sebagainya. Menurut Tacbir, data mining adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan machine learning untuk mengekstraksi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari database yang besar. Istilah data mining memiliki hakikat sebagai disiplin ilmu yang tujuan utamanya adalah untuk menemukan, menggali, atau menambang pengetahuan dari data atau informasi yang kita miliki. Proses menggali informasi dalam data mining melibatkan integrasi teknik dari berbagai disiplin ilmu, seperti teknologi database dan data warehouse, statistik, machine learning, komputasi dengan kinerja tinggi, pattern recognition, neural network, visualisasi data dan sebagainya.

Data mining menggunakan pendekatan discovery-based dimana pencocokan pola (pattern matching) dan algoritma-algoritma yang lain digunakan untuk menentukan relasi-relasi kunci di dalam data yang dieksplorasi. Data mining (penambangan data), sesuai dengan namanya, berkonotasi sebagai pencarian informasi bisnis yang berharga dari basis data yang sangat besar. Dengan tersedianya basis data dalam kualitas dan ukuran yang memadai, teknologi data mining memiliki kemampuan-kemampuan sebagai berikut:

1. Mengotomatisasi prediksi trend sifat-sifat bisnis. Data mining mengotomatisasi proses pencarian informasi di dalam basis data yang besar.
2. Mengotomatisasi penemuan pola-pola yang tidak diketahui sebelumnya. Tools data mining "menyapu" basis data, kemudian mengidentifikasi pola-pola yang sebelumnya tersembunyi dalam satu sapuan. Contoh dari penemuan pola ini adalah analisis pada data penjualan ritel untuk mengidentifikasi produk-produk yang kelihatannya tidak berkaitan, yang seringkali dibeli secara bersamaan oleh customer.

## Tahapan dalam *Data Mining*

Sebagai suatu rangkaian proses, *data mining* dapat dibagi menjadi beberapa tahap proses yang diilustrasikan pada Gambar 3 Tahap-tahap tersebut bersifat interaktif, pemakai terlibat langsung atau dengan perantaraan *knowledge base*.



Gambar 3 Tahap-tahap *Data Mining*

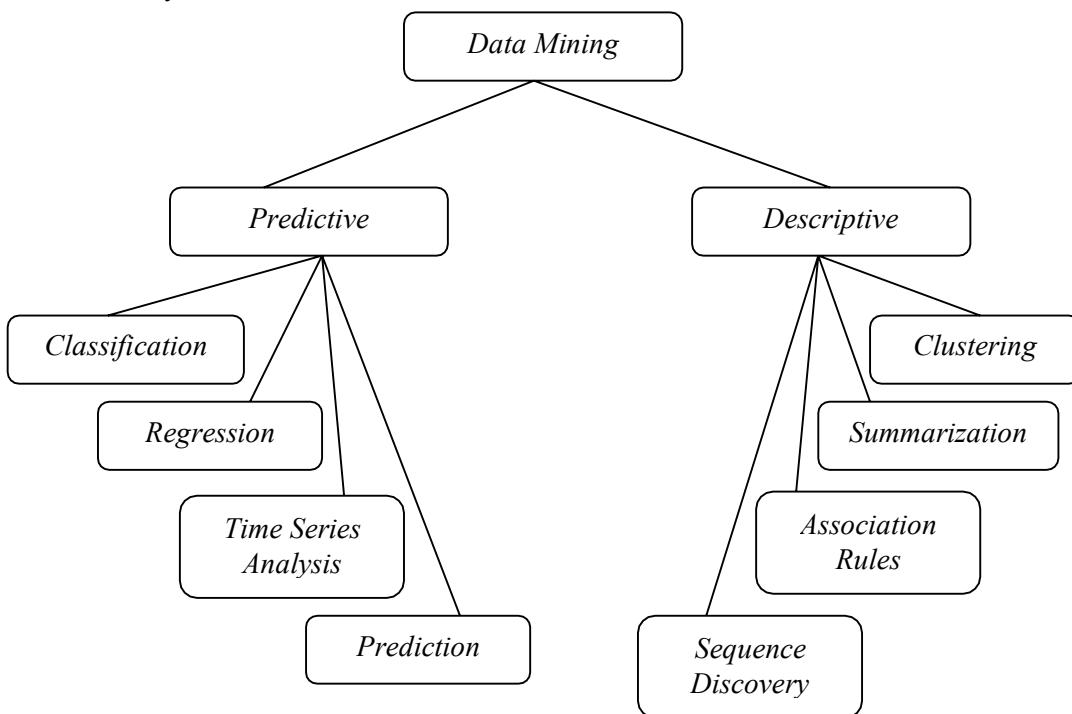
Tahap-tahap *data mining* adalah sebagai berikut:

- a. Pembersihan data (*data cleaning*)  
Pembersihan data merupakan proses menghilangkan *noise* dan data yang tidak konsisten atau data tidak relevan.
- b. Integrasi data (*data integration*)  
Integrasi data merupakan penggabungan data dari berbagai *database* ke dalam satu *database* baru.
- c. Seleksi data (*data selection*)  
Data yang ada pada *database* sering kali tidak semuanya dipakai, oleh karena itu hanya data yang sesuai untuk dianalisis yang akan diambil dari *database*.
- d. Transformasi data (*data transformation*)  
Data diubah atau digabung ke dalam format yang sesuai untuk diproses dalam *data mining*.
- e. Proses *mining*  
Merupakan suatu proses utama saat metode diterapkan untuk menemukan pengetahuan berharga dan tersembunyi dari data.
- f. Evaluasi pola (*pattern evaluation*)  
Untuk mengidentifikasi pola-pola menarik ke dalam *knowledge based* yang ditemukan.
- g. Presentasi pengetahuan (*knowledge presentation*)  
Merupakan visualisasi dan penyajian pengetahuan mengenai metode yang digunakan untuk memperoleh pengetahuan yang diperoleh pengguna.

## Teknik-Teknik *Data mining*

*Data mining* adalah serangkaian proses untuk menggali nilai tambah dari suatu kumpulan data berupa pengetahuan yang selama ini tidak diketahui secara manual. Perlu diingat bahwa kata *mining* sendiri berarti usaha untuk mendapatkan sedikit data berharga dari sejumlah besar data dasar. Karena itu *data mining* sebenarnya memiliki akar yang panjang dari bidang ilmu seperti kecerdasan buatan (*artificial intelligent*), *machine learning*, statistik dan basis data.

Menurut Ahmed, teknik *data mining* biasanya terbagi dalam dua kategori, prediksi dan deskripsi. Teknik prediksi menggunakan data historis untuk menyimpulkan sesuatu tentang kejadian di masa depan. Sedangkan teknik deskripsi bertujuan untuk menemukan pola dalam data yang menyediakan beberapa informasi tentang hubungan interval yang tersembunyi.



Menurut Kumar dan Saurabh, terdapat beberapa teknik yang digunakan dalam *data mining*, yaitu:

### 1. *Classification*

Klasifikasi adalah teknik yang paling umum diterapkan pada *data mining*. Pendekatan ini sering menggunakan keputusan pohon (*decision tree*) atau *neural network* berbasis algoritma klasifikasi. Proses klasifikasi data melibatkan *learning* dan klasifikasi. Dalam belajar (*learning*) data pelatihan (*training*) dianalisis dengan algoritma klasifikasi. Dalam klasifikasi pengujian data dilakukan dengan menggunakan perkiraan akurasi dari aturan klasifikasi. Jika akurasi bisa diterima, maka aturan dapat diterapkan untuk data baru. Salah satu contoh yang mudah dan populer adalah dengan *decision tree* yaitu salah satu metode klasifikasi yang paling populer karena mudah untuk diinterpretasi. *Decision tree* adalah model prediksi menggunakan struktur pohon atau struktur berhirarki.

*Decision tree* adalah struktur *flowchart* yang menyerupai *tree* (pohon), dimana setiap simpul internal menandakan suatu tes pada atribut, setiap cabang merepresentasikan hasil tes, dan simpul daun merepresentasikan kelas atau distribusi kelas. Alur pada *decision tree* di telusuri dari simpul akar ke simpul daun yang memegang prediksi kelas untuk contoh tersebut. *Decision tree* mudah untuk dikonversi ke aturan klasifikasi (*classification rules*).

## 2. *Clustering*

*Clustering* bisa dikatakan sebagai identifikasi kelas objek yang memiliki kemiripan. Dengan menggunakan teknik *clustering* kita bisa lebih lanjut mengidentifikasi kepadatan dan jarak daerah dalam objek ruang dan dapat menemukan secara keseluruhan pola distribusi dan korelasi antara atribut. Pendekatan klasifikasi secara efektif juga dapat digunakan untuk membedakan kelompok atau kelas objek.

## 3. *Predication*

Teknik regresi dapat disesuaikan untuk prediksi. Analisis regresi dapat digunakan untuk model hubungan antara satu atau lebih *independent variables* dan *dependent variables*. Dalam *data mining* *independent variabel* adalah atribut-atribut yang sudah dikenal dan respon variabel apa yang kita inginkan untuk diprediksi. Akan tetapi, banyak masalah di dunia nyata bukan prediksi yang mudah. Karena itu, teknik kompleks (seperti: *logistic regression*, *decision trees* atau pohon keputusan, *neural nets* atau jaringan syaraf) mungkin akan diperlukan untuk memprediksi nilai. Model yang berjenis sama sering dapat digunakan untuk regresi dan klasifikasi. Misalnya, *CART (Classification and Regression Trees)* yaitu algoritma pohon keputusan yang dapat digunakan untuk membangun kedua pohon klasifikasi dan pohon regresi. Jaringan saraf juga dapat menciptakan kedua model klasifikasi dan regresi.

## 4. *Association rule*

Digunakan untuk mengenali kelakuan dari kejadian-kejadian khusus atau proses dimana link asosiasi muncul pada setiap kejadian. Contoh dari aturan assosiatif dari analisa pembelian di suatu pasar swalayan adalah bisa diketahui berapa besar kemungkinan seorang pelanggan membeli roti bersamaan dengan susu. Dengan pengetahuan tersebut pemilik pasar swalayan dapat mengatur penempatan barangnya atau merancang kampanye pemasaran dengan memakai kupon diskon untuk kombinasi barang tertentu.

Penting tidaknya suatu aturan assosiatif dapat diketahui dengan dua parameter, *support* yaitu prosentasi kombinasi atribut tersebut dalam basisdata dan *confidence* yaitu kuatnya hubungan antar atribut dalam aturan assosiatif. Motivasi awal pencarian *association rule* berasal dari keinginan untuk menganalisa data transaksi supermarket, ditinjau dari perilaku *customer* dalam membeli produk. *Association rule* ini menjelaskan seberapa sering suatu produk dibeli secara bersamaan. Sebagai contoh, *association rule* “*beer => diaper (80%)*” menunjukkan bahwa empat dari lima *customer* yang membeli *beer* juga membeli *diaper*. Dalam suatu *association rule*  $X \Rightarrow Y$ ,  $X$  disebut dengan *antecedent* dan  $Y$  disebut dengan *consequent rule*.

## 5. *Neural network*

Jaringan saraf adalah seperangkat unit penghubung *input* dan *output* dimana setiap konesinya memiliki bobot. Selama fase *learning*, jaringan belajar dengan menyesuaikan bobot sehingga dapat memprediksi kelas yang benar label dari setiap *input*. Jaringan saraf memiliki kemampuan yang luar biasa untuk memperoleh arti

dari data yang rumit atau tidak tepat dan dapat digunakan untuk mengambil pola-pola serta mendeteksi *tren* yang sangat kompleks untuk diperhatikan baik oleh manusia atau teknik komputer lain. Jaringan saraf sangat baik untuk mengidentifikasi pola atau *tren* pada data dan sangat cocok untuk melakukan prediksi serta memprediksi kebutuhan.

6. *Decision trees*

*Decision trees* atau pohon keputusan adalah struktur *tree-shaped* yang mewakili set keputusan. Keputusan ini menghasilkan aturan untuk klasifikasi sebuah kumpulan data. Metode pohon keputusan diantaranya yaitu *Classification and regression trees* (CART) dan *Chi Square Automatic Interaction Detection* (CHAID).

7. *Nearest Neighbor Method*

Teknik yang mengklasifikasikan setiap *record* dalam sebuah kumpulan data berdasarkan sebuah kombinasi suatu kelas  $k$  *record* yang sama dalam sebuah kumpulan data historis (dimana  $k$  lebih besar atau sama dengan 1). Terkadang disebut juga dengan teknik *K-Nearest Neighbor*.

## Clustering

Madhu Yedha mendefenisikan *clustering* sebagai proses pengorganisasian objek data ke dalam *set* kelas yang saling berhubungan, yang disebut *cluster*. *Clustering* merupakan contoh dari klasifikasi tanpa arahan (*unsupervised*). Klasifikasi merujuk kepada prosedur yang menetapkan objek data set kelas. *Unsupervised* berarti bahwa pengelompokan tidak tergantung pada standar kelas dan pelatihan atau *training*.

Menurut Deka, *Clustering* merupakan salah satu teknik *data mining* yang digunakan untuk mendapatkan kelompok-kelompok dari objek-objek yang mempunyai karakteristik yang umum di data yang cukup besar. Tujuan utama dari metode *clustering* adalah pengelompokan sejumlah data atau objek ke dalam *cluster* atau grup sehingga dalam setiap *cluster* akan berisi data yang semirip mungkin. *Clustering* melakukan pengelompokan data yang didasarkan pada kesamaan antar objek, oleh karena itu klasterisasi digolongkan sebagai metode *unsupervised learning*. Menurut Oyelade, *clustering* dapat dibagi menjadi dua, yaitu *hierarchical clustering* dan *non-hierarchical clustering*.

*Hierarchical clustering* adalah suatu metode pengelompokan data yang dimulai dengan mengelompokkan dua atau lebih objek yang memiliki kesamaan paling dekat. Kemudian proses diteruskan ke objek lain yang memiliki kedekatan kedua. Demikian seterusnya sehingga *cluster* akan membentuk semacam pohon dimana ada hierarki (tingkatan) yang jelas antar objek, dari yang paling mirip sampai yang paling tidak mirip. Secara logika semua objek pada akhirnya hanya akan membentuk sebuah *cluster*. *Dendogram* biasanya digunakan untuk membantu memperjelas proses hierarki tersebut.

Berbeda dengan metode *hierarchical clustering*, metode *non-hierarchical clustering* justru dimulai dengan menentukan terlebih dahulu jumlah *cluster* yang diinginkan (dua *cluster*, tiga *cluster*, atau lain sebagainya). Setelah jumlah *cluster* diketahui, baru proses *cluster* dilakukan tanpa mengikuti proses hierarki. Metode ini biasa disebut dengan *K-Means Clustering*.

### Algoritma *K-means Clustering*

*K-means clustering* merupakan salah satu metode *cluster analysis* non hirarki yang berusaha untuk mempartisi objek yang ada kedalam satu atau lebih *cluster* atau

kelompok objek berdasarkan karakteristiknya, sehingga objek yang mempunyai karakteristik yang sama dikelompokan dalam satu *cluster* yang sama dan objek yang mempunyai karakteristik yang berbeda dikelompokan kedalam *cluster* yang lain.

Menurut Daniel dan Eko, Langkah-langkah algoritma *K-Means* adalah sebagai berikut:

- a. Pilih secara acak  $k$  buah data sebagai pusat *cluster*.
- b. Jarak antara data dan pusat *cluster* dihitung menggunakan *Euclidian Distance*. Untuk menghitung jarak semua data ke setiap titik pusat *cluster* dapat menggunakan teori jarak *Euclidean* yang dirumuskan sebagai berikut:
$$D(i,j) = \sqrt{(X_{1i} - X_{1j})^2 + (X_{2i} - X_{2j})^2 + \dots + (X_{ki} - X_{kj})^2}$$
dimana:

$D(i,j)$	= Jarak data ke $i$ ke pusat <i>cluster</i>
$j X_{ki}$	= Data ke $i$ pada atribut data ke $k$
$X_{kj}$	= Titik pusat ke $j$ pada atribut ke $k$
- c. Data ditempatkan dalam *cluster* yang terdekat, dihitung dari tengah *cluster*.
- d. Pusat *cluster* baru akan ditentukan bila semua data telah ditetapkan dalam *cluster* terdekat.
- e. Proses penentuan pusat *cluster* dan penempatan data dalam *cluster* diulangi sampai nilai *centroid* tidak berubah lagi.

## Contoh Soal

Untuk meningkatkan pemahaman kita, mari kita bahas contoh soal berikut lengkap dengan perhitungannya. Dimisalkan kita memiliki sampel data dalam tabel berikut. Ada 6 buah data yang akan kita kelompokkan menjadi 2 cluster. Kita sebut saja K1 dan K2.

Sample Data	X	Y	Kelompok / Cluster
1	100	50	
2	40	60	
3	30	70	
4	90	10	
5	65	40	
6	25	35	

contoh kelompok data untuk menghitung K-Means

Pertama kita akan menghitung centroid. Kita ambil data ke-1 dan ke-2 sebagai perhitungan pertama. Kita menggunakan rumus Euclidean Distance untuk mendapatkan jarak minimum data terhadap centroid.

Cluster	X	Y
K1	100	50
K2	40	60

penggunaan data ke-1 dan ke-2

Berikut ini adalah rumus dari Euclidean Distance.

$$[(x, y), (a, b)] = \sqrt{(x - a)^2 + (y - b)^2}$$

persamaan Euclidean Distance

### Perhitungan Pertama

Kita mulai menghitung centroid pertama. Kita akan menentukan jarak dari data 1 ke data 1, data 1 ke data 2, data 2 ke data 1 dan data 2 ke data 2.

$$\text{Cluster 1 } (100, 50) = \sqrt{(100 - 100)^2 + (50 - 50)^2} = 0$$

(jarak cluster 1 ke cluster 1)

Jarak dari Cluster 2 ke cluster 1 (40 , 60)  $\leftrightarrow$  (100,50) =

$$\sqrt{(40 - 100)^2 + (60 - 50)^2} = \sqrt{(60)^2 + (10)^2} = \sqrt{3600 + 100} = \sqrt{3700} = 60.83$$

Jarak dari cluster 1 ke cluster 2 (100, 50)  $\leftrightarrow$  (40, 60) =

$$\sqrt{(100 - 40)^2 + (50 - 60)^2} = \sqrt{(60)^2 + (-10)^2} = \sqrt{3600 + 100} = \sqrt{3700} = 60.83$$

$$\text{Jarak cluster 2 ke cluster 2 } (40, 60) = \sqrt{(40 - 40)^2 + (60 - 60)^2} = 0$$

menghitung cluster 1 dan 2 untuk data ke-1 dan data ke-2

Pada bagian yang diberikan garis merah adalah data dan perhitungan, sedangkan yang diberikan garis biru adalah hasil yang akan kita masukkan ke dalam tabel untuk menentukan data tersebut akan masuk kedalam cluster K1 atau K2. Sehingga kita dapatkan hasil seperti berikut :

Cluster	Centroid		Kelompok Cluster
	X	Y	
K1 (100, 50)	0	60.83	1
K2 (40, 60)	60.83	0	2

hasil pengelompokan cluster data ke-1 masuk K1 dan data ke-2 masuk K2

Dari tabel diatas, kita lihat jarak minimum dari data 1 ke data 1 adalah 0 dan jarak minimum dari data 2 ke data 2 yaitu 0. Sehingga yang menjadi centroid K1 adalah data 1 dan data 2 menjadi centroid K2. Selanjutnya kita beralih ke perhitungan kedua untuk data ke-3.

## Perhitungan Kedua

Setelah mendapatkan centroid, kita beralih ke data ke-3 yaitu (30, 70). Kita mulai hitung jarak data ke-3 terhadap centroid 1 dan centroid 2. Sehingga hasilnya nanti kita mengetahui data 3 akan masuk ke cluster K1 atau K2. Berikut adalah perhitungannya.

Langkah selanjutnya kita beralih ke data 3 yaitu (30, 70). Kita mulai menghitung jarak dataset terhadap cluster 1.

Centroid  
Dataset

$$(100, 50) \leftrightarrow (30, 70) =$$

$$\sqrt{(30 - 100)^2 + (70 - 50)^2} = \sqrt{(-70)^2 + (20)^2} = \sqrt{4900 + 400} = \sqrt{5300} = 72.80$$

Kemudian kita hitung jarak dataset terhadap cluster 2.

$$(40, 60) \leftrightarrow (30, 70) =$$

$$\sqrt{(30 - 40)^2 + (70 - 60)^2} = \sqrt{(-10)^2 + (10)^2} = \sqrt{100 + 100} = \sqrt{200} = 14.14$$

perhitungan untuk data ke-3

Garis hijau sebagai centroid, garis merah sebagai dataset dan garis biru sebagai hasil. Sebenarnya perhitungan  $(x-a)^2$  dan  $(y-b)^2$  dapat dibalik menjadi  $(a-x)^2$  dan  $(b-y)^2$  karena hasilnya pasti positif sebab dikuadratkan.

Semuanya dihitung dengan Euclidean Distance dan hasilnya kita masukkan dalam tabel berikut.

Dataset	Euclidean Distance		Kelompok Cluster
	Cluster 1	Cluster 2	
(30, 70)	72.80	14.14	2

hasil perhitungan data ke-2

Dilihat dari jarak minimum data terhadap centroid, data ke-3 lebih dekat ke K2 dengan nilai 14.14.

Selanjutnya kita meng-UPDATE nilai Centroid. Karena data masuk ke K2, maka centroid K2 diupdate dengan cara :

$$\begin{aligned} X_{\text{centroid\_baru}} &= (x_{\text{K2}} + x_{\text{data3}})/2 \\ Y_{\text{centroid\_baru}} &= (y_{\text{K2}} + y_{\text{data3}})/2 \end{aligned}$$

Sehingga centroid yang baru kita dapatkan , pada garis merah adalah persamaan perhitungannya dan pada kotak merah adalah hasilnya.

Cluster	X	Y
K1	100	50
K2	$= \frac{40 + 30}{2} = 35$	$= \frac{60 + 70}{2} = 65$

Jadi Cluster Centroid yang baru adalah sebagai berikut :

Cluster	X	Y
K1	100	50
K2	35	65

hasil update centroid untuk K2

## Perhitungan Ketiga

Berlanjut ke data ke-4 yaitu (90, 10), kita mulai menghitung jarak antar dataset dan Centroid K1, sama seperti langkah di perhitungan kedua. dan berikut adalah perhitungannya.

$$(100, 50) \leftrightarrow (90, 10) =$$

$$\sqrt{(90 - 100)^2 + (10 - 50)^2} = \sqrt{(-10)^2 + (-40)^2} = \sqrt{100 + 1600} = \sqrt{1700} = 41.23$$

Perhitungan dataset ke-4 dengan K1

Kemudian kita hitung jarak antar dataset dan Centroid K2.

Perlu diingat, karena K2 telah diupdate, jadi kita harus menggunakan nilai centroid K2 yang baru yaitu : (35, 65). Hal ini juga berpengaruh apabila Centroid K1 diupdate.  
Berikut ini adalah perhitungannya.

$$(35, 65) \leftrightarrow (90, 10) =$$

Dataset  
centroid baru

$$\sqrt{(90 - 35)^2 + (10 - 65)^2} = \sqrt{(55)^2 + (-55)^2} = \sqrt{3025 + 3025} = \sqrt{6050} = 77.78$$

perhitungan dataset ke-4 dengan K2 yang telah diupdate

Dari kedua perhitungan jarak dataset ke K1 dan K2, didapatkan hasil sebagai berikut :

Dataset	Euclidean Distance		Kelompok Cluster
	Cluster 1	Cluster 2	
(90, 10)	41.23	77.78	1

hasil perhitungan data ke-4 masuk ke K1

Dari hasil diatas, kita mendapatkan jika dataset ke-4 masuk dalam cluster 1. Jadi seperti langkah sebelumnya, kita update kembali centroid K1 dengan dataset ke-3.

Kemudian kita update Centroid

Cluster	X	Y
K1	$= \frac{100 + 90}{2} = 95$	$= \frac{50 + 10}{2} = 30$
K2	35	65

Jadi Cluster Centroid yang baru adalah sebagai berikut :

Cluster	X	Y
K1	95	30
K2	35	65

update Centroid K1

Perhitungan selanjutnya untuk dataset ke-5 dan ke-6 mengikuti langkah-langkah yang telah kita lakukan pada dataset ke-3 dan ke-4. Yang perlu diperhatikan adalah menggunakan centroid yang telah di update dan juga melakukan update centroid yang baru.

Dari hasil perhitungan hingga dataset ke-6 didapatkan hasil seperti tabel berikut :

Sample Data	X	Y	Kelompok / Cluster
1	100	50	1
2	40	60	2
3	30	70	2
4	90	10	1
5	65	40	1
6	25	35	2

hasil akhir perhitungan K-Means

Sumber Link :

- <https://www.ketutrare.com/2018/11/algoritma-k-means-clustering-dan-contoh.html>
- <https://ilmukomputer.org/wp-content/uploads/2018/05/agus-k-means-clustering.pdf>
- <https://www.neliti.com/id/publications/195857/penerapan-data-mining-untuk-clustering-data-penduduk-miskin-menggunakan-algoritm>
- <https://media.neliti.com/media/publications/103765-ID-clustering-menggunakan-metode-k-mean-unt.pdf>
- <https://www.jagoanhosting.com/blog/apa-itu-data-mining/>

## RINGKASAN TUTORIAL MENGIKUR JARAK DAN LUAS SUATU AREAL MENGGUNAKAN ARCGIS

Link tutorial : <https://www.youtube.com/watch?v=mBIxKf2uTD4>

Aplikasi yang digunakan yaitu arcgis 10, kordinat sistem menggunakan WGS 1964 dengan study kasus yaitu menghitung dari citra satelit dan menggunakan bing satelite, dengan menghitung panjang landasan bandara tabing kota padang. Dengan langkah awal adalah membuat sop baru untuk jenis panjang yang digunakan yaitu line coordinate system menggunakan yang UTM ZONE 475. Dari menu editor kita klik menu create features, lalu drawing line pada bagian yang ingin kita ukur lalu klik kanan finish sketch. Untuk mengetahui suatu jarak atau lokasi pada bandara yang kita ukur yaitu klik kanan pd menu project lalu klik label ganti dengan jarak. Panjang landasannya adalah 2140 meter.

Selanjutnya menghitung suatu luas dari areal bandara tabing kota padang, yg kita lakukan adalah klik catalog, arcgis lalu new shapefile dengan type polygon dengan nama polygon bandara. Coordinate systemnya UTM Zone 475 lalu klik ok. Lalu klik menu create features dan star editing. Lalu klik batas-batas yang ingin kita hitung suatu luasnya lalu klik kanan finish editing. Cara menghitung luasnya yaitu klik kanan polygon bandara klik atribut table lalu add field lalu ketikkan luas lalu ok. Untung mengetahui luasnya kita ubah satuan menjadi hectares lalu ok. Luas dari komplek bandara adalah 247 hectares.

Sekian terimakasih..

## TUTORIAL PENGUKURAN JARAK ANTAR DATA

Untuk melakukan pengukuran jarak antar data sebaiknya kita mengetahui jenis – jenis data. Sehingga kita tidak mengalami kesulitan dalam mengelompokkan data berdasarkan jenis dan tipe datanya.

### Macam macam Data

Dalam data data mining dan maha datar, Anda akan menemukan banyak jenis data yang berbeda, dan masing-masing cenderung membutuhkan alat dan teknik yang berbeda. Macam macam data dikelompokkan sebagai berikut:

- *Data terstruktur (structured)*
- *Data tidak terstruktur(unstructured)*
- *Data bahasa alami(Natural Language)*
- *Data yang dibangkit oleh Mesin (Machined-Generated)*
- *Data Audio, Video, Citra*
- *Data Streamming*
- *Data berbasis Graph(Graph-based)*

### Data Terstruktur

Data terstruktur adalah data yang bergantung pada model data dan yang dinyatakan dalam bentuk tabel dengan atribut} (kolom) dan baris. Data terstruktur mudah disimpan dalam database dalam bentuk tabel atau file excel (Ms Office), SQL (structure Query Language) sehingga mudah dilakukan query terhadap data tersebut. Tetapi realitanya banyak data yang ada dalam dalam bentuk data tidak terstruktur karena data dihasilkan oleh manusia dan mesin

### Macam- macam atribut

Atribut adalah data yang mewakili karakteristik atau fitur dari objek data. Atribut bisa disebut juga dengan dimensi, fitur, dan variabel yang istilah itu sering digunakan literatur. Dimensi istilah yang biasanya digunakan dalam data warehouse. Dalam literatur pembelajaran mesin cenderung menggunakan istilah fitur, sementara dalam bidang statistik lebih memilih menggunakan istilah variabel. Dalam penambangan data atau data mining dan database biasa menggunakan istilah atribut atau fitur , dan dalam buku ini juga menggunakan istilah atribut atau fitur. Contoh atribut-atribut yang menggambarkan objek pelanggan dapat mencakup, misalnya ID pelanggan, nama, dan alamat. Nilai yang diamati untuk atribut tertentu dikenal sebagai nilai observasi. Sekumpulan atribut yang digunakan untuk menggambarkan objek disebut disebut dengan vektor atribut (atau vektor fitur). Distribusi data yang melibatkan satu atribut (atau variabel) disebut univariat. Distribusi bivariat melibatkan dua atribut, dan seterusnya. Jenis atribut ditentukan oleh nilai-nilai pada atribut tersebut yang mungkin nominal, biner, atau ordinal, atau numerik. Pada subbagian berikut, kami perkenalkan nilai nilai tersebut.

Macam macam tipe data atribut

- *Atribut Nominal*

Nilai atribut nominal adalah simbol atau nama barang. Setiap nilai mewakili beberapa jenis kategori, kode, atau status, dan Atribut nominal juga disebut kategori. Nilai-nilainya tidak memiliki tingkatan nilai. Dalam ilmu komputer, nilainya juga dikenal sebagai enumerasi.

**Contoh :**

Misalkan warna rambut dan status perkawinan adalah dua atribut dari data orang. Nilai yang mungkin untuk warna rambut adalah hitam, coklat, pirang, merah, hitam pucat, abu-abu, dan putih. Status perkawinan memiliki nilai atribut lajang, menikah, bercerai, dan janda. Baik warna rambut maupun status perkawinan adalah atribut nominal. Contoh lain dari atribut nominal adalah atribut pekerjaan dengan nilai-nilainya adalah guru, dokter gigi, programmer, petani, dan sebagainya.

- *Atribut Biner*

Atribut biner adalah atribut nominal dengan hanya dua kategori atau status: 0 atau 1, di mana 0 biasanya berarti atribut itu tidak ada, dan 1 berarti itu ada. Atribut Biner disebut sebagai Boolean jika dinyatakan dengan benar (true) dan salah(false)

**Contoh :**

Terdapat atribut yang menggambarkan merokok pada pasien, 1 menunjukkan bahwa pasien merokok, sementara 0 menunjukkan bahwa pasien tidak merokok. Demikian pula, seandainya ada pasien menjalani tes medis yang memiliki dua kemungkinan hasil. Atribut Tes medis bersifat biner, dengan nilai 1 berarti hasil tes untuk pasien positif, sedangkan 0 berarti hasilnya negatif. Atribut biner simetris jika keduanya memiliki nilai bobot yang sama; Artinya, tidak ada kekhususan mengenai hasil mana yang harus dikodekan sebagai 0 atau 1. Misalkan atribut gender yang dengan nilai atributnya laki dan perempuan. Atribut biner adalah asimetris jika hasil dari nilai nilainya tidak sama pentingnya seperti hasil positif dan negatif dari tes medis untuk HIV. Dengan mengkodekan hasil yang paling penting, biasanya 1 (mis., HIV positif) dan yang lainnya dengan 0 (mis., HIV negatif)

- *Atribut ordinal*

Atribut ordinal adalah atribut dengan nilai yang memiliki arti urutan atau peringkat di antara nilai-nilai yang ada, tapi besarnya nilai yang berurutan tersebut tidak diketahui. Ukuran kecenderungan terpusat dari atribut ordinal dapat diwakili oleh modus dan median (nilai tengah), tetapi tidak untuk nilai rata-rata. Perlu diperhatikan bahwa atribut nominal, biner, dan ordinal bersifat kualitatif. Artinya, atribut-atribut tersebut hanya menjelaskan sebuah fitur dari suatu objek tanpa memberikan ukuran atau kuantitas yang sebenarnya. Nilai-nilai atribut kualitatif biasanya merupakan kata-kata yang mewakili kategori

**Contoh:**

Atribut ordinal Misalkan ukuran minuman yang tersedia di sebuah restoran cepat saji. Atribut nominal ini memiliki tiga nilai yang mungkin: kecil, sedang, dan besar. Nilai memiliki arti

urutan yang (yang sesuai dengan ukuran minuman). Contoh atribut ordinal lainnya adalah pangkat dan jabatan profesi. Atribut ordinal berguna untuk melakukan penilaian subjektif terhadap kualitas sesuatu objek yang tidak dapat diukur secara obyektif; atribut ordinal sering digunakan dalam survei untuk peringkat. Dalam satu survei, para peserta diminta untuk menilai tingkat kepuasan mereka sebagai pelanggan. Kepuasan pelanggan memiliki kategori ordinal berikut ini: 0: sangat tidak puas, 1: agak tidak puas, 2: netral, 3: puas, dan 4: sangat puas. Atribut ordinal juga dapat diperoleh dari iskritisasi nilai atribut numerik dengan membagi rentang nilai menjadi urutan kategoris

- *Atribut Numerik*

Atribut numerik bersifat kuantitatif; Artinya, ini adalah kuantitas yang terukur, yang dinyatakan dengan bilangan bulat atau nilai riel. Atribut numerik dapat Atribut Skala Interval(interval-scaled) atau skala ration (ratio-scaled)

- **Atribut skala interval** diukur pada dengan skala unit ukuran yang sama. Nilai - nilai Interval berskala memiliki urutan dan bisa positif, 0, atau negatif. Jadi, selain untuk memberikan peringkat nilai, atribut semacam itu memungkinkan kita untuk membandingkan dan mengukur perbedaan antar nilai

**Contoh:**

Atribut suhu adalah Skala interval. Misalkan kita memiliki nilai suhu di luar ruangan untuk beberapa hari yang berbeda dari suatu objek. Dengan mengurutkan nilai, kita mendapatkan peringkat objek yang berkenaan dengan suhu. Selain itu, kita bisa mengukur perbedaan antara nilai. Misalnya, a suhu 20°C adalah lima derajat lebih tinggi dari suhu 15°C. Contoh lain kalender tahun adalah. Misalnya, tahun 2002 dan 2010 terpisah delapan tahun. Karena atribut skala interval adalah numerik, kita dapat menghitung nilai ratarata, ukuran median dan modus dari kecenderungan terpusat

- **Atribut Skala Ratio** Atribut skala rasio adalah atribut numerik dengan melekat titik nol pada nilai atribut tersebut. Artinya, jika pengukuran adalah berskala rasio, kita dapat dapat mengatakan berapa kali dari nilai yang lain atau rasio dari nilai yang lain. Selain itu, nilai yang dipesan, dan kita juga bisa menghitung selisih antara nilai, serta mean, median, dan modus

**Contoh**

Atribut tentang pengukuran berat badan, tinggi badan, jumlah kata dalam dokumen

### **Data Tidak Terstruktur**

Data tidak terstruktur adalah data yang tidak mudah dimasukkan ke dalam model data karena isi/kontennya spesifik atau bervariasi. Salah satu contoh data tidak terstruktur adalah data email. Meskipun email berisi elemen terstruktur seperti pengirim, judul, dan isi teks, terlalu banyak variasi dari isi yang terkandung dalamnya diantaranya dialek bahasa yang dipakai dan sebagainya. Email juga salah satu contoh data bahasa alami

## **Bahasa Alami**

Dalam neuropsikologi , linguistik , dan filsafat bahasa , bahasa alami atau bahasa biasa adalah bahasa yang telah berevolusi secara alami pada manusia melalui penggunaan dan pengulangan tanpa perencanaan. Bahasa alami berbeda dengan bahasa yang dibangun untuk memprogramma komputer atau membangun logika nalar. Bahasa alami dikenal sebagai bahasa manusia misal bahasa indonesia, bahasa inggris dan lain lain. Didalam pemrosesan bahasa alami diperluangkan pengetahuan ilmu linguistics, semantics, statistics and machine learning.Dengan pemrosesan bahasa alami membantu komputer untuk memahami bahasa yang telah diucapkan oleh manusia.

## **Data yang dibangkitkan oleh Mesin**

Data yang dibangkitkan oleh mesin secara otomatis tanpa intervensi manusia. Data ini terus menerus dibangkitkan selama proses tertentu sedang berjalan. Misalkan data weblog dari mesin server yang dihasilkan dari hasil transaksi user dengan sistem web. Contoh lain adalah data yang dihasilkan dari implementasi internet of things misal perekaman suhu udara dan kelembaban udara dari daerah tertentu yang terhubung dengan pusat penyimpanan data tersebut.

## **Data jaringan atau data berbasis Graph**

Data graph adalah data yang dinyatakan dengan graph yang dalam matematika mengacu pada konsep teori graph. Data ini menunjukkan keterhubungan antara objek objek atau relasi antar objek objek dengan menggunakan struktur node, edge, dan karakteristik/sifat keterhubungan antar objek tersebut. Salah satu data graph adalah data keterhubungan orang dalam media sosial. Dengan memanfaatkan data graph media sosial kita dapat mengukur ukuran ukuran tertentu berdasarkan struktur yang dibentuknya. Misalkan menentukan pengaruh orang dalam struktur jaringan tersebut, apakah termasuk orang penting/berpengaruh atau bukan. Gambar berikut menunjukkan contoh data graph. Database graph dapat digunakan untuk menyimpan data berbasis graph dan menggunakan query tertentu yaitu SPARQL

## **Data Audio,Vidio dan Citra**

Dengan perkembangan teknologi implementasi multimedia yang sangat pesat saat,data audio,video dan citra cukup besar dihasilkan dari transaksi bisnis. Dengan besarnya data yang dihasilkan membutuhkan proses pengolahan spesifik dari data tersebut untuk dimanfaatkan terutama dalam analisa data sain. Diantara pemanfaatan data multimedai tersebut adalah pengenalan objek, pengenala suara, segmentasi citra satelit dan banyak analisa lain yang dihasilkan dari data multimedai tersebut.

## **Data streaming**

Data Streaming adalah data yang dihasilkan secara terus-menerus oleh ribuan sumber data, yang biasanya mengirimkan catatan data secara bersamaan, dan dalam ukuran kecil (urutan Kilobyte). Data streaming mencakup berbagai macam data seperti logfile yang dihasilkan oleh pelanggan aplikasi seluler atau website Anda, transaksi e-commerce, informasi dari jejaring sosial, data geospasial, dan perangkat sensor yang terhubung atau instrumentasi di pusat data.

Data ini perlu diproses secara berurutan dan bertahap secara record-by-record digunakan untuk berbagai macam analisis misalkan korelasi, agregasi, penyaringan, dan pengambilan sampel. Informasi yang diperoleh dari analisis tersebut memberikan petunjuk terhadap pelanggan mereka seperti penggunaan layanan mereka, aktivitas server, klik website, dan lain lain. Misalnya, dalam bisnis kita dapat melacak perubahan sentimen publik pada merek dan produk mereka dengan menganalisis aliran data media sosial, sehingga dapat merespons secara tepat baik waktu dan tindakan yang harus dilakukan

## Distribusi Data

Karakteristik utama dari data adalah distribusi probabilitasnya. Distribusi data yang paling dikenal adalah distribusi normal atau Gaussian. Distribusi ini ditemukan pada sistem fisik dimana data dibangkitkan secara acak. Fungsi dinyatakan dalam bentuk fungsi padat probabilitas(probability density function)

$$f(x) = \frac{1}{(\sigma\sqrt{2\pi})} \frac{e^{-(x-\mu)^2}}{(2\sigma^2)}$$

Dimana  $\sigma$  adalah standar deviasi dan  $\mu$  adalah mean. Persamaan ini menyatakan peluang variable acak dari suatu data x

Kita menyatakan standar deviasi sebagai lebar kurva lonceng dan rata rata sebagai pusat. Kadangkala istilah variance digunakan dan ini adalah kuadrat dari standar deviasi. Standar deviasi pada dasarnya mengukur bagaimana sebaran data.

Untuk memahami lebih jelasnya bagaimana fungsi tersebut digambarkan, berikut implementasinya data dengan distribusi normal yang memiliki rata-rata 1 dan variansinya 0.5

## Statistik Deskriptif

### Ukuran Kecenderungan Terpusat

#### Rata-rata (Mean)

Pada bagian ini, kami melihat cara untuk mengukur kecenderungan pusat data. Misalkan kita mempunyai atribut hasil pretest yang dinyatakan dengan atribut X. Misalkan  $x_1, x_2, \dots, x_N$

menjadi himpunan nilai N yang diamati atau pengamatan untuk X. Di sini, nilai-nilai ini juga dapat disebut set data (untuk X). Jika kita merencanakan pengamatan untuk nilai pretest, di mana sebagian besar nilai berada? Ini memberi kita gambaran tentang kecenderungan pusat dari data. Ukuran kecenderungan pusat data ukurannya adalah rata-rata(mean), median, modus (mode), dan midrange. Atribut numerik yang paling umum dan efektif dari "pusat" dari set data adalah mean (aritmatika). Misalkan  $x_1, x_2, \dots, x_N$

menjadi satu set nilai N atau pengamatan, Rata-rata dari nilai pretes dinyatakan dengan

$$\bar{x} = \frac{\sum_{i=1}^N x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}$$

Kadang-kadang, setiap nilai  $x_i$

dalam satu data dapat dikaitkan dengan bobot  $w_i$  untuk  $i=1,..,N$

. Bobot tersebut mencerminkan signifikansi, kepentingan, atau frekuensi kejadian yang melekat pada masing masing nilai. Dalam hal ini, kita dapat menghitungnya dengan

-

$$\bar{x} = \frac{\sum_{i=1}^N w_i x_i}{\sum_{i=1}^N w_i} = \frac{w_1 x_1 + w_2 x_2 + \cdots + w_N x_N}{w_1 + w_2 + \cdots + w_N}$$

Meskipun rata-rata adalah jumlah yang sangat berguna untuk menggambarkan kumpulan data, itu tidak selalu cara terbaik untuk mengukur pusat data. Masalah utama dengan mean adalah sensitivitasnya terhadap nilai ekstrim (mis., outlier). Bahkan beberapa nilai ekstrem saja dapat merusak mean. Misalnya, gaji rata-rata di suatu perusahaan mungkin sangat besar didorong oleh beberapa manajer bergaji tinggi. Demikian pula, nilai rata-rata kelas di ujian dapat rata-rata rendah karena beberapa ada beberapa skor nilai saja yang sangat rendah. Untuk mengimbangi efek tersebut kita bisa menggunakan rata-rata yang dipangkas (trimmed mean), yang merupakan rata-rata yang diperoleh setelah memangkas nilai paling tinggi dan nilai yang paling rendah. Untuk contoh, kita dapat mengurutkan nilai gaji yang diamati kemudian menghapus 2% atas dan bawah nilai tersebut sebelum menghitung mean. Kita harus menghindari pemotongan bagian yang terlalu besar (seperti 20%) pada kedua ujungnya, karena hal ini dapat mengakibatkan hilangnya informasi yang berharga)

### **Median**

Untuk data miring (asimetris), ukuran pusat data yang lebih baik adalah median, yang merupakan nilai tengah dalam satu set nilai data yang diurutkan. Ini adalah nilai yang memisahkan separuh data yang lebih tinggi dari data tersebut dan sebagian data yang lebih rendah dari data tersebut. Dalam probabilitas dan statistik, median umumnya berlaku untuk data numerik; namun, kami dapat memperluas konsep menjadi data ordinal. Misalkan kumpulan N data yang diberikan untuk atribut X diurutkan dalam urutan naik. Jika N ganjil, maka median adalah nilai tengah dari data yang ordinal. Jika N adalah genap, maka mediannya tidak unik; dihitung dengan rata rata dari nilai  $\left(\frac{N}{2}+1\right) + \left(\frac{N}{2}-1\right)$

$$Me = x_{ij} + \left( \frac{\frac{n}{2} - f_{kij}}{f_i} \right) p$$

Namun pada data berkelompok, dengan data yang berbentuk kelas interval, kita tidak bisa langsung mengetahui nilai median jika kelas mediannya sudah diketahui dengan formula

$$Me = \text{median}$$

$$x_{ij} = \text{batas bawah median}$$

$$n = \text{jumlah data}$$

$$f_{kij} = \text{frekuensi kumulatif data di bawah kelas median}$$

$$f_i = \text{frekuensi data pada kelas median}$$

$$p = \text{panjang interval kelas}$$

Contoh :

**Mode** adalah ukuran lain dari kecenderungan sentral. Mode (modus) untuk satu set data adalah nilai yang paling sering terjadi di set. Oleh karena itu, dapat ditentukan untuk atribut kualitatif dan kuantitatif. Dimungkinkan untuk frekuensi terbesar untuk bersesuaian beberapa nilai berbeda, yang menghasilkan lebih dari satu mode. Kumpulan data dengan satu, dua, atau tiga mode masing-masing disebut unimodal, bimodal, dan trimodal. Jika data hanya mengandung nilai data terjadi hanya sekali, maka tidak ada modus

Untuk data numerik unimodal yang cukup miring (asimetris), kami memiliki hubungan empiris:

$$\text{mean} - \text{mode} \approx 3 \times (\text{mean} - \text{median})$$

Ini menyiratkan bahwa mode untuk kurva frekuensi unimodal yang cukup miring dapat dengan mudah didekati jika nilai rata-rata dan median diketahui.

## Mengukur Sebaran Data

Kita sekarang membahas ukuran ukuran untuk menilai dispersi atau penyebaran data numerik. Ukuran-ukuran itu adalah rentang (range), kuartil, kuartil, persentil, dan rentang interkuartil. Semua itu adalah ringkasan lima angka, yang dapat ditunjukkan dengan boxplot, berguna dalam mengidentifikasi pencikan (outlier). Varians dan standar deviasi juga menunjukkan sebaran distribusi data.

### **Rentang (Range), Quartil, and Rentang Interquartile**

Misalkan  $x_1, x_2, \dots, x_N$

adalah sekumpulan pengamatan untuk atribut numerik,  $X$

. Rentang adalah selisih antara nilai terbesar (maks ()) dan terkecil (min ()). Misalkan data untuk atribut  $X$  diurutkan dalam urutan naik. Bagilah data berdasarkan titik-titik tertentu sehingga membagi distribusi data ukuran yang sama, seperti pada Gambar dibawah. Titik data ini disebut kuartil. 2-quantile adalah titik data yang membagi bagian bawah dan atas dari distribusi data. Ini sama dengan median. 4-kuantil adalah tiga titik data yang membagi distribusi data menjadi empat bagian yang sama; setiap bagian mewakili seperempat dari distribusi data. Ini lebih sering disebut sebagai kuartil. 100-kuantil lebih sering disebut

sebagai persentil; mereka membagi distribusi data menjadi 100 data berukuran sama. Median, kuartil, dan persentil adalah bentuk kuantil yang paling banyak digunakan.

Kuartil memberikan gambaran pusat distribusi, penyebaran, dan bentuk distribusi. Kuartil satu, dilambangkan oleh Q1, adalah persentil ke-25. Nilai ini menunjukkan 25% terendah dari data. Kuartil ketiga, dilambangkan oleh Q3, adalah persentil ke-75 - itu memisahkan data 75% dari terendah data (atau 25% dari tertinggi data). Kuartil kedua adalah persentil ke-50 atau median dari distribusi data.

Jarak antara kuartil pertama dan ketiga adalah ukuran yang menyatakan rentang yang dicakup oleh bagian tengah data. Jarak ini disebut rentang interkuartil (IQR) dan dinyatakan dengan

$$IQR = Q_3 - Q_1$$

Dengan ukuran a kuartil Q1 dan Q3, dan median kita dapat mengidentifikasi ada tidaknya penculan (outlier) pada suatu data. Data penculan atau outlier nilai data biasanya ada di setidaknya  $1,5 \times IQR$  di atas kuartil ketiga atau di bawah kuartil pertama

Karena Q1, median, dan Q3 tidak berisi informasi tentang titik akhir (mis., Ekor) data, ringkasan yang lebih lengkap dari bentuk distribusi dapat diperoleh dengan memberikan nilai data terendah dan tertinggi juga. Ini dikenal sebagai ringkasan lima angka. Ringkasan lima nomor distribusi terdiri dari median (Q2), kuartil Q1 dan Q3, dan data terkecil dan terbesar( Minimum, Q1, Median, Q3, Maksimum)

Boxplots adalah cara populer untuk memvisualisasikan distribusi. Boxplot menggabungkan ringkasan lima angka sebagai berikut: - Ujung kotak adalah kuartil dan panjang kotak adalah rentang interkuartil. - Median ditandai dengan garis di dalam kotak. - Dua garis (disebut whiskers) di luar kotak memanjang ke pengamatan terkecil (Minimum) dan terbesar (Maksimum)

Outlier biasanya ada di dibawah  $Q_1 - 1.5 \times IQR$

dan diatas  $Q_3 + 1.5 \times IQR$

### ***Variansi dan Standar Deviasi***

Variansi dan standar deviasi adalah ukuran penyebaran data. Nilai-nilai tersebut menunjukkan bagaimana penyebaran distribusi data. Standar Deviasi yang rendah berarti bahwa pengamatan data cenderung sangat dekat dengan rata-rata, sedangkan deviasi standar yang tinggi menunjukkan data tersebar di sejumlah nilai-nilai besar.

Varian dari pengamatan  $N, x_1, x_2, \dots, x_N$  untuk atribut numerik X adalah

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 = \left( \frac{1}{N} \sum_{i=1}^N x_i^2 \right) - \bar{x}^2$$

di mana  $\overline{x}$  adalah nilai rata-rata dari pengamatan, Standar deviasi,  $\sigma$ , dari pengamatan adalah akar kuadrat dari variansi,  $\sigma^2$

Sifat dasar dari standar deviasi,  $\sigma$  sebagai ukuran penyebaran data adalah sebagai berikut:

- Ukuran  $\sigma$  mengeukur sebaran disekitar rata-rata dan harus dipertimbangkan bila rata-rata dipilih sebagai ukuran pusat data
- $\sigma=0$  hanya jika tidak ada penyebaran data, hanya terjadi ketika semua pengamatan memiliki nilai sama, Jika tidak maka  $\sigma>0$

### **Skewness**

Derajat distorsi dari kurva lonceng simetris atau distribusi normal. Ini mengukur kurangnya simetri dalam distribusi data Untuk menghitung derajat distorsi dapat menggunakan Koefisien Kemencengan Pearson yang diperoleh dengan menggunakan nilai selisih rata-rata dengan modus dibagi simpangan baku. Koefisien Kemencengan Pearson dirumuskan sebagai berikut :

$$sk = \frac{\overline{X} - Mo}{s}$$

dengan

$$\overline{X} - Mo \approx 3(\overline{X} - Me)$$

maka

$$sk \approx \frac{3(\overline{X} - Me)}{s}$$

### **Implementasi**

Untuk implementasi silahkan unduh [data.csv](#)

```
import pandas as pd
from scipy import stats
df=pd.read_csv("data.csv",usecols=[0])

print("jumlah data ",df['NilaiPreTest'].count())
print("rata-rata ",df['NilaiPreTest'].mean())
print("nila minimal ",df['NilaiPreTest'].min())
print("Q1      ",df['NilaiPreTest'].quantile(0.25))
```

```

print("Q2      ",df['NilaiPreTest'].quantile(0.5))
print("Q3      ",df['NilaiPreTest'].quantile(0.75))
print("Nilai Max  ",df['NilaiPreTest'].max())
print("kemencengan","{0:.2f}".format(round(df['NilaiPreTest'].skew(),2)))
mode=stats.mode(df)
print("Nilai modus {} dengan jumlah {}".format(mode.mode[0], mode.count[0]))
print("kemencengan      ","{0:.6f}".format(round(df['NilaiPreTest'].skew(),6)))
print("Standar Deviasi  ","{0:.2f}".format(round(df['NilaiPreTest'].std(),2)))
print("Variansi      ","{0:.2f}".format(round(df['NilaiPreTest'].var(),2)))

```

## Mengukur Jarak Data

### Mengukur Jarak Tipe Numerik

**Shirkhorshidi, A. S., Aghabozorgi, S., & Wah, T. Y. (2015). A comparison study on similarity and dissimilarity measures in clustering continuous data. PloS one, 10(12), e0144059.**

Salah satu tantangan dalam era ini dengan database yang memiliki banyak tipe data. Mengukur jarak adalah komponen utama dalam algoritma clustering berbasis jarak. Alogritma seperit Algoritma Partisioning misal K-Mean, K-medoidm dan fuzzy c-mean dan rough clustering bergantung pada jarak untuk melakukan pengelompokan

Sebelum menjelaskan tentang beberapa macam ukuran jarak, kita mendefinisikan terlebih dahulu yaitu v1,v2

menyatakan dua vektor yang menyatakan  $v1 = x_1, x_2, \dots, x_n, v2 = y_1, y_2, \dots, y_n$ , dimana  $x_i, y_i$

disebut attribut. Ada beberapa ukuran similaritas data atau ukuran jarak, diantaranya

#### *Minkowski Distance*

Kelompok Minkowski diantaranya adalah Euclidean distance dan Manhattan distance, yang menjadi kasus khusus dari Minkowski distance. Minkowski distance dinyatakan dengan

$$d_{\min} = (\sum_{i=1}^n |x_i - y_i|^m)^{\frac{1}{m}}, m \geq 1$$

dimana  $m$  adalah bilangan riil positif dan  $x_i$  dan  $y_i$  adalah dua vektor dalam ruang dimensi  $n$

Implementasi ukuran jarak Minkowski pada model clustering data atribut dilakukan normalisasi untuk menghindari dominasi dari atribut yang memiliki skala data besar.

### ***Manhattan distance***

Manhattan distance adalah kasus khusus dari jarak Minkowski distance pada  $m = 1$ . Seperti Minkowski Distance, Manhattan distance sensitif terhadap outlier. Bila ukuran ini digunakan dalam algoritma clustering , bentuk cluster adalah hyper-rectangular. Ukuran ini didefinisikan dengan

$$d_{\text{man}} = \sum_{i=1}^n |x_i - y_i|$$

### ***Euclidean distance***

Jarak yang paling terkenal yang digunakan untuk data numerik adalah jarak Euclidean. Ini adalah kasus khusus dari jarak Minkowski ketika  $m = 2$ . Jarak Euclidean berkinerja baik ketika digunakan untuk kumpulan data cluster kompak atau terisolasi . Meskipun jarak Euclidean sangat umum dalam pengelompokan, ia memiliki kelemahan: jika dua vektor data tidak memiliki nilai atribut yang sama, kemungkinan memiliki jarak yang lebih kecil daripada pasangan vektor data lainnya yang mengandung nilai atribut yang sama. Masalah lain dengan jarak Euclidean sebagai fitur skala terbesar akan mendominasi yang lain. Normalisasi fitur kontinu adalah solusi untuk mengatasi kelemahan ini.

### ***Average Distance***

Berkenaan dengan kekurangan dari Jarak Euclidian Distance diatas, rata rata jarak adalah versi modifikasi dari jarak Euclidian untuk memperbaiki hasil. Untuk dua titik  $x,y$

dalam ruang dimensi  $n$ , rata-rata jarak didefinisikan dengan

$$d_{\text{ave}} = \left( \frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

### ***Weighted euclidean distance***

Jika berdasarkan tingkatan penting dari masing masing atribut ditentukan, maka Weighted Euclidean distance adalah modifikasi lain dari jarak Euclidean distance yang dapat digunakan. Ukuran ini dirumuskan dengan

$$d_{we} = \left( \sum_{i=1}^n w_i (x_i - y_i) \right)^2 )^{\frac{1}{2}}$$

dimana  $w_i$  adalah bobot yang diberikan pada atribut ke i.

### ***Chord distance***

Chord distance adalah salah satu ukuran jarak modifikasi Euclidean distance untuk mengatasi kekurangan dari Euclidean distance. Ini dapat dipecahkan juga dengan menggunakan skala pengukuran yang baik. Jarak ini dapat juga dihitung dari data yang tidak dinormalisasi . Chord distance didefinisikan dengan

$$d_{\text{chord}} = \left( 2 - 2 \frac{\sum_{i=1}^n x_i y_i}{\|x\|_2 \|y\|_2} \right)^{\frac{1}{2}}$$

dimana  $\|x\|_2$  adalah  $L^2$ -norm  $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$

### ***Mahalanobis distance***

Mahalanobis distance berdasarkan data berbeda dengan Euclidean dan Manhattan distances yang bebas antara data dengan data yang lain. Jarak Mahalanobis yang teratur dapat digunakan untuk mengekstraksi hyperellipsoidal clusters. Jarak Mahalanobis dapat mengurangi distorsi yang disebabkan oleh korelasi linier antara fitur dengan menerapkan transformasi pemutihan ke data atau dengan menggunakan kuadrat Jarak mahalanobis. Mahalanobis distance dinyatakan dengan

$$d_{\text{mah}} = \sqrt{(x - y) S^{-1} (x - y)^T}$$

dimana S adalah matrik covariance data.

### **Cosine measure**

Ukuran Cosine similarity lebih banyak digunakan dalam similaritas dokumen dan dinyatakan dengan

$$\text{Cosine}(x, y) = \frac{\sum_{i=1}^n x_i y_i}{\|x\|_2 \|y\|_2}$$

dimana  $\|y\|_2$  adalah Euclidean norm dari vektor  $y = (y_1, y_2, \dots, y_n)$  didefinisikan dengan

$$\|y\|_2 = \sqrt{y_1^2 + y_2^2 + \dots + y_n^2}$$

### **Pearson correlation**

Pearson correlation banyak digunakan dalam data expresi gen. Ukuran similaritas ini menghitung similaritas antara dua bentuk pola expresi gen. Pearson correlation didefinisikan dengan

$$\text{Pearson}(x, y) = \frac{\sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

The Pearson correlation kelebihannya adalah sensitif terhadap outlier

### **Mengukur Jarak Atribut Binary**

Mari kita lihat similaritas dan desimilarity untuk objek yang dijelaskan oleh atribut biner simetris atau asimetris. Atribut biner hanya memiliki dua status: 0 dan 1. Contoh atribut perokok menggambarkan seorang pasien, misalnya, 1 menunjukkan bahwa pasien merokok, sedangkan 0 menunjukkan pasien tidak merokok. Memperlakukan atribut biner sebagai atribut numerik tidak diperkenankan. Oleh karena itu, metode khusus untuk data biner diperlukan untuk membedakan komputasi.

Jadi, bagaimana kita bisa menghitung ketidaksamaan antara dua atribut biner? "Satu pendekatan melibatkan penghitungan matriks ketidaksamaan dari data biner yang diberikan. Jika semua atribut biner dianggap memiliki bobot yang sama, kita memiliki tabel kontingensi  $2 \times 2$

di mana q adalah jumlah atribut yang sama dengan 1 untuk kedua objek i dan j, r adalah jumlah atribut yang sama dengan 1 untuk objek i tetapi 0 untuk objek j, s adalah jumlah atribut yang sama dengan 0 untuk objek i tetapi 1 untuk objek j, dan t adalah jumlah atribut yang sama dengan 0 untuk kedua objek i dan j. Jumlah total atribut adalah p, di mana  $p = q + r + s + t$

Ingatlah bahwa untuk atribut biner simetris, masing-masing nilai bobot yang sama. Dissimilarity yang didasarkan pada atribut asymmetric binary disebut symmetric binary dissimilarity. Jika objek i dan j dinyatakan sebagai atribut biner simetris, maka dissimilarity antar i dan j adalah

$$d(i, j) = \frac{r + s}{q + r + s + t}$$

Untuk atribut biner asimetris, kedua kondisi tersebut tidak sama pentingnya, seperti hasil positif (1) dan negatif (0) dari tes penyakit. Diberikan dua atribut biner asimetris, pencocokan keduanya 1 (kecocokan positif) kemudian dianggap lebih signifikan daripada kecocokan negatif. Ketidaksamaan berdasarkan atribut-atribut ini disebut asimetris biner dissimilarity, di mana jumlah kecocokan negatif, t, dianggap tidak penting dan dengan demikian diabaikan. Berikut perhitungannya

$$d(i, j) = \frac{r + s}{q + r + s}$$

Kita dapat mengukur perbedaan antara dua atribut biner berdasarkan pada dissimilarity. Misalnya, biner asimetris kesamaan antara objek i dan j dapat dihitung dengan

$$\text{sim}(i, j) = \frac{q}{q + r + s} = 1 - d(i, j)$$

Persamaan similarity ini disebut dengan **Jaccard coefficient**

### Mengukur Jarak Tipe categorical

*Li, C., & Li, H. (2010). A Survey of Distance Metrics for Nominal Attributes. JSW, 5(11), 1262-1269.*

#### *Overlay Metric*

Ketika semua atribut adalah bertipe nominal, ukuran jarak yang paling sederhana adalah dengan Overlay Metric (OM) yang dinyatakan dengan

$$d(x, y) = \sum_{i=1}^n \delta(a_i(x), a_i(y))$$

dimana n adalah banyaknya atribut,  $a_i(x)$  dan  $a_i(y)$  adalah nilai atribut ke i yaitu  $A_i$  dari masing-masing objek x dan y,  $\delta(a_i(x), a_i(y))$  adalah 0 jika  $a_i(x) = a_i(y)$  dan 1 jika sebaliknya.

OM banyak digunakan oleh instance-based learning dan locally weighted learning. Jelas sekali, ini sedikit beruk untuk mengukur jarak antara masing-masing pasangan sample, karena gagal memanfaatkan tambahan informasi yang diberikan oleh nilai atribut nominal yang bisa membantu dalam generalisasi.

### ***Value Difference Metric (VDM)***

VDM dikenalkan oleh Standfill and Waltz, versi sederhana dari VDM tanpa skema pembobotan didefinisikan dengan

$$d(x, y) = \sum_{i=1}^n \sum_{c=1}^C |P(c|a_i(x)) - P(c|a_i(y))|$$

dimana C adalah banyaknya kelas,  $P(c|a_i(x))$  adalah probabilitas bersyarat dimana kelas x adalah c dari atribut  $A_i$ , yang memiliki nilai  $a_i(x)$ ,  $P(c|a_i(y))$  adalah probabilitas bersyarat dimana kelas y adalah c dengan atribut  $A_i$  memiliki nilai  $a_i(y)$

VDM mengasumsikan bahwa dua nilai dari atribut adalah lebih dekat jika memiliki klasifikasi sama. Pendekatan lain berbasis probabilitas adalah SFM (Short and Fukunaga Metric) yang kemudian dikembangkan oleh Myles dan Hand dan didefinisikan dengan

$$d(x, y) = \sum_{c=1}^C |P(c|x) - P(c|y)|$$

dimana probabilitas keanggotaan kelas diestimasi dengan  $P(c|x)$  dan  $P(c|y)$  didekati dengan Naive Bayes,

### ***Minimum Risk Metric (MRM)***

Ukuran ini dipresentasikan oleh Blanzieri and Ricci, berbeda dari SFM yaitu meminimumkan selisih antara kesalahan berhingga dan kesalahan asymptotic. MRM meminimumkan risk of misclassification yang didefinisikan dengan

$$d(x, y) = \sum_{c=1}^C P(c|x)(1 - P(c|y))$$

## Mengukur Jarak Tipe Ordinal

**Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.**

Nilai-nilai atribut ordinal memiliki urutan atau peringkat, namun besarnya antara nilai-nilai berturut-turut tidak diketahui. Contohnya tingkatan kecil, sedang, besar untuk atribut ukuran. Atribut ordinal juga dapat diperoleh dari diskritisasi atribut numerik dengan membuat rentang nilai ke dalam sejumlah kategori tertentu. Kategori-kategori ini disusun dalam peringkat. Yaitu, rentang atribut numerik dapat dipetakan ke atribut ordinal f

yang memiliki Mf state. Misalnya, kisaran suhu atribut skala-skala (dalam Celcius) dapat diatur ke dalam status berikut: -30 hingga -10, -10 hingga 10, 10 hingga 30, masing-masing mewakili kategori suhu dingin, suhu sedang, dan suhu hangat. M adalah jumlah keadaan yang dapat dilakukan oleh atribut ordinal memiliki. State ini menentukan peringkat 1,...,Mf

Perlakuan untuk atribut ordinal adalah cukup sama dengan atribut numerik ketika menghitung dissimilarity antara objek. Misalkan f

adalah atribut-atribut dari atribut ordinal dari n

objek. Menghitung dissimilarity terhadap f fitur sebagai berikut:

- Nilai f untuk objek ke-i adalah xif, dan f memiliki Mf status urutan , mewakili peringkat 1,...,Mf Ganti setiap xif dengan peringkatnya, rif $\in\{1...M_f\}$
- Karena setiap atribut ordinal dapat memiliki jumlah state yang berbeda, diperlukan untuk memetakan rentang setiap atribut ke [0,0, 1.0] sehingga setiap atribut memiliki bobot yang sama. Perlakukan normalisasi data dengan mengganti peringkat rif dengan zif=rif-1Mf-1

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

- Dissimilarity kemudian dihitung dengan menggunakan ukuran jarak seperti atribut numerik dengan data yang baru setelah ditransformasi \$ z\_{\\_if} \$

## Menghitung Jarak Tipe Campuran

**Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. Journal of artificial intelligence research, 6, 1-34.**

Menghitung ketidaksamaan antara objek dengan atribut campuran yang berupa nominal, biner simetris, biner asimetris, numerik, atau ordinal yang ada pada kebanyakan databasae dapat dinyatakan dengan memproses semua tipe atribut secara bersamaan. Salah satu teknik

tersebut menggabungkan atribut yang berbeda ke dalam matriks ketidaksamaan tunggal dan menyatakannya dengan skala interval antar [0,0,1.0]

. Misalkan data berisi atribut p tipe campuran. Ketidaksamaan (disimilarity ) antara objek i dan j

dinyatakan dengan

$$d(i, j) = \frac{\sum_{f=1}^p \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^p \delta_{ij}^{(f)}}$$

dimana  $\delta_{ij}^f = 0$  - jika  $x_{if}$  atau  $x_{jf}$  adalah hilang (i.e., tidak ada pengukuran dari atribut f untuk objek i atau objek j)

- jika  $x_{if} = x_{jf} = 0$  dan
- atribut f adalah binary asymmetric,

selain itu  $\delta_{ij}^f = 1$

Kontribusi dari atribut f untuk dissimilarity antara i dan j (yaitu. $d_{ij}^f$ ) dihitung bergantung pada tipenya,

- Jika f adalah numerik,  $d_{ij}^f = \frac{\|x_{if} - x_{jf}\|}{\max_h x_{hf} - \min_h x_{hf}}$ , di mana h menjalankan semua nilai objek yang tidak hilang untuk atribut f
- Jika f adalah nominal atau binary,\$d\_{ij}^f = 0\$ jika  $x_{if} = x_{jf}$ , sebaliknya  $d_{ij}^f = 1$
- Jika f adalah ordinal maka hitung rangking  $r_{if}$  dan  $z_{if} = \frac{r_{if}-1}{M_f-1}$  , dan perlakukan  $z_{if}$  sebagai numerik.

## PENGUKURAN JARAK ANTAR DATA

### CLUSTERING

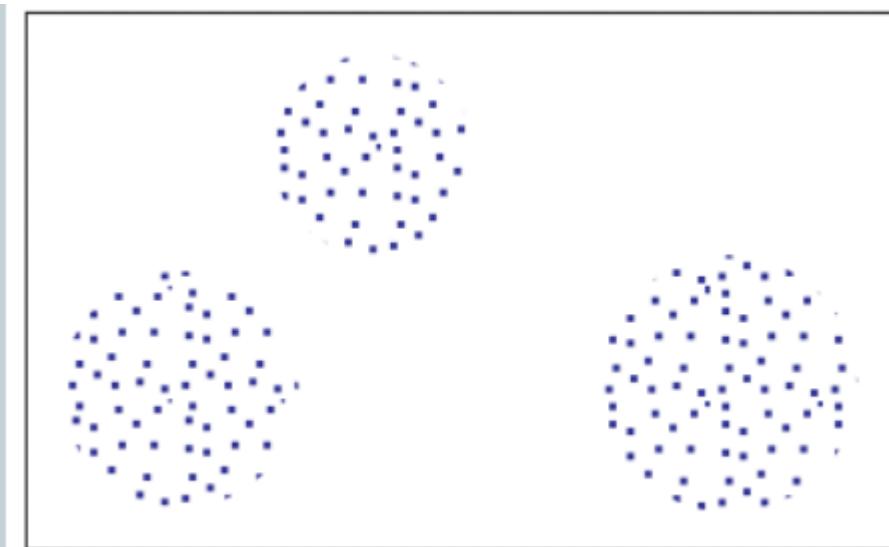
Secara umum cluster didefinisikan sebagai “sejumlah objek yang mirip yang dikelompokan secara bersama”, Namun definisi dari cluster bisa beragam tergantung dari sudut pandang yang digunakan, beberapa definisi cluster berdasarkan sudut pandang adalah sebagai berikut:

#### Definisi Well-Separated Cluster

Berdasarkan definisi ini cluster adalah sekelompok titik (objek) dimana sebuah titik pada kelompok itu lebih dekat atau mirip dengan semuanya (objek) yang ada pada kelompok tersebut dari pada titik-titik (objek-objek) lain yang tidak terdapat pada kelompok itu.

Biasanya digunakan sebuah nilai batas (threshold) untuk menentukan titik-titik (objek-objek) yang dianggap cukup dekat satu samalainnya.

Namun terdapat kelemahan pada definisinya itu titik-titik yang terdapat pada “pojok” sebuah cluster pada kenyataannya mungkin saja lebih dekat dengan titik-titik pada cluster yang lain.

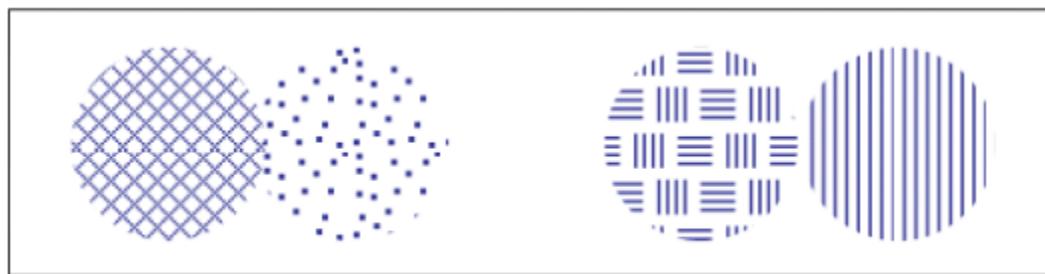


**Cluster berdasarkan definisi Well-Separated-Cluster**

#### Definisi Center-Based Cluster

Berdasarkan definisi ini sebuah cluster didefinisikan sebagai sekelompok titik (objek) dimana semua titik pada kelompok itu lebih dekat dengan pusat atau “center” dari kelompok tersebut dari pada pusat pada kelompok lainnya.

Umumnya pusat cluster adalah centroid, yaitu rata-rata dari semua titik pada cluster tersebut, namun dapat juga digunakan medoid,yaitu titik yang paling mewakili pada sebuah cluster.



**Cluster berdasarkan definisi Center-Based Cluster**

### Cluster Analysis

Cluster analysis merupakan salah satu metode Data mining yang bersifat tanpa latihan (unsupervised analisys) yang mempunyai tujuan untuk mengelompokan data kedalam kelompok-kelompok dimana data yang berada dalam kelompok yang sama akan mempunyai sifat yang relatif homogen.

Jika ada n objek pengamatan dengan p variable maka terlebih dulu ditentukan ukuran kedekatan sifat antar data, ukuran kedekatan sifat data yang bisa digunakan adalah jarak euclidian (Euclidean distance) antara dua objek dari p dimensi pengamatan, jika objek pertama yang akan diamati adalah  $X = [x_1, x_2, x_3, \dots, x_p]$  dan  $Y = [y_1, y_2, y_3, \dots, y_p]$  maka euclidean distance dirumuskan sebagai berikut:

$$d_{XY} = \sqrt{\sum_{j=1}^p (x_j - y_j)^2}$$

Secara formal definisi dari cluster analysis adalah sebagai berikut:

Misalkan S adalah himpunan objek yang mempunyai n buah elemen,

$S = \{o_1, o_2, o_3, \dots, o_n\}$  (II.1)

Cluster analysis membagi S (didefinisikan pada persamaan II.1) menjadi k himpunan  $C_1, C_2, C_3, \dots, C_k$ , himpunan-himpunan tersebut disebut dengan cluster. Sebuah cluster  $C_i$  adalah subset atau himpunan bagian dari S, Solusi atau keluaran dari sebuah cluster Analysis dinyatakan sebagai himpunan dari semua cluster,

Jika S adalah himpunan objek yang mempunyai n buah elemen dan terdiri dari r variable maka ketika S dibagi menjadi k cluster, maka model dari cluster dapat didefinisikan dengan dua buah matrik yaitu matrik data.

$D_{n \times k} = (d_{ik})$  dan matrik variable  $F_{r \times k} = (f_{jk})$ ,

$$d_{ik} = \begin{cases} 1, & \text{data ke } i \text{ anggota kluster ke } k \\ 0, & \text{data ke } i \text{ bukan anggota kluster ke } k \end{cases}$$

Proses clustering mengasumsikan bahwa data akan menjadi anggota dari satu dan hanya satu cluster.

$$f_{jk} = \begin{cases} 1, & \text{Variable ke } j \text{ anggota kluster ke } k \\ 0, & \text{Variable ke } j \text{ bukan anggota kluster ke } k \end{cases}$$

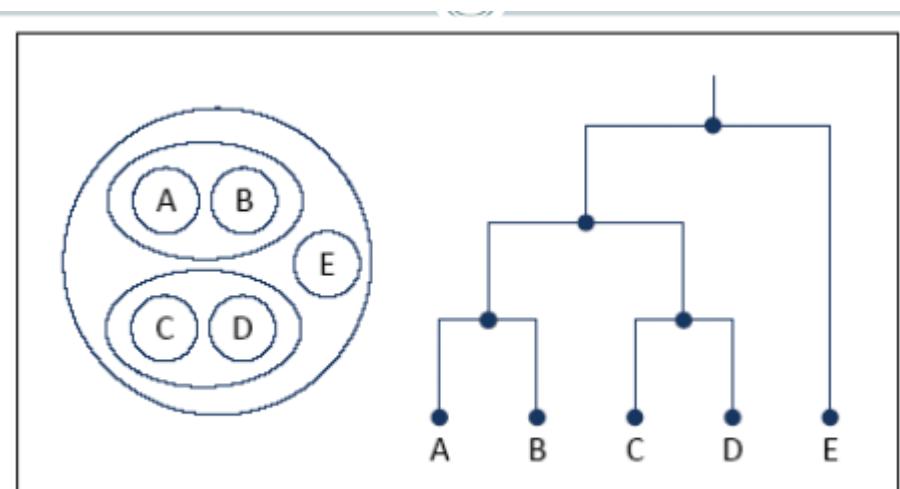
## Klasifikasi Metode Cluster Analysis

Metode cluster analysis pada dasarnya ada dua jenis, yaitu metode cluster analysis hirarki (hierarchical clustering method) dan Metode cluster analysis non hirarki (non hierarchical clustering method).

### Metode clustering hirarki

Metode clustering hirarki digunakan apabila belum ada informasi jumlah cluster yang akan dipilih, metode hirarki akan menghasilkan cluster-cluster yang bersarang (nested) sehingga masing-masing cluster dapat memiliki sub-cluster.

Prinsip utama metode cluster analysis hirarki adalah mengatur semua objek dalam sebuah pohon keputusan (umumnya berupa pohon biner) berdasarkan suatu fungsi kriteria tertentu. Pohon tersebut disebut dendogram.



**Contoh Dendogram**

Semakin tinggi level simpul pohon maka semakin rendah tingkat similaritas antar objeknya, metode cluster analysis hirarki dapat dilakukan dengan dua pendekatan yaitu bottom-up (agglomerative) dan top-down (divisive).

Pada pendekatan agglomerative setiap objek pada awalnya berada pada cluster masing-masing, kemudian setiap cluster yang paling mirip akan dikelompokan dalam satu cluster, hingga membentuk suatu hirarki cluster.

Pada pendekatan divisive, pada awalnya hanya terdapat satu buah cluster tunggal yang beranggotakan seluruh objek, kemudian dilakukan pemecahan atas cluster tersebut menjadi beberapa sub-cluster,

Contoh algoritma metode cluster hirarki adalah HAC (Hierarchical Agglomerative Clustering) dengan beberapa variasi perhitungan similaritas antar cluster seperti single-link, complete-link dan group average.

### **Metode Cluster Analysis Non Hirarki**

Metode cluster analysis non hirarki biasa juga disebut dengan partitional clustering bertujuan mengelompokan objek ke dalam k cluster ( $k < n$ ) dimana nilai k sudah ditentukan sebelumnya.

Salah satu prosedur clustering non hirarki adalah menggunakan metode K-Means clustering analisis, yaitu metode yang bertujuan untuk mengelompokan objek atau data sedemikian rupa sehingga jarak tiap objek kepusat cluster (centroid) adalah minimum, titik pusat cluster terbentuk dari rata-rata nilai dari setiap variable.

Secara umum proses cluster analysis dimulai dengan perumusan masalah clustering dengan mendefinisikan variable-variable yang akan digunakan sebagai dasar proses cluster.

Konsep dasar dari cluster analysis adalah konsep pengukuran jarak(distance) atau kesamaaan (similarity),

- Distance adalah ukuran tentang jarak pisah antar objek sedangkan similaritas adalah ukuran kedekatan.
- Pengukuran jarak (distance type measure) digunakan untuk data-data yang bersifatmetrik, sedangkan pengukuran kesesuaian (matching type measure) digunakan untuk data-data yang bersifat kualitatif atau non metrik.
- Proses clustering yang baik seharusnya menghasilkan cluster-cluster yang berkualitas tinggi dengan sifat-sifat sebagai berikut:
  1. Setiap objek pada cluster memiliki kemiripan (intra cluster similarity) yang tinggi satu sama lainnya.
  2. Kemiripan objek pada cluster yang berbeda (inter cluster similarity) rendah.

### **K-Means Cluster Analysis**

K-means cluster analysis merupakan salah satu metode cluster analysis non hirarki yang berusaha untuk mempartisi data yang ada kedalam satu atau lebih cluster atau kelompok data berdasarkan karakteristiknya, sehingga data yang mempunyai karakteristik yang sama dikelompokan dalam satu cluster yang sama dan data yang mempunyai karakteristik yang berbeda dikelompokan kedalam cluster yang lain.

Tujuannya adalah untuk meminimalkan objective function yang di set dalam proses clustering,yang pada dasarnya berusaha untuk meminimalkan variasi dalam satu cluster dan memaksimalkan variasi antar cluster.

K-Means meliputi sequential threshold, pararel threshold dan optimizing threshold. Sequential threshold melakukan pengelompokan dengan terlebih dahulu memilih satu objek dasar yang akan dijadikan nilai awal cluster, kemudian semua cluster yang ada dalam jarak terdekat dengan cluster ini akan bergabung, lalu dipilih cluster kedua dan semua objek yang mempunyai kemiripan dengan cluster ini akan digabungkan, demikian seterusnya sehingga terbentuk beberapa cluster dengan keseluruhan objek terdapat didalamnya.

**Pararel threshold** secara prinsip sama dengan sequential threshold hanya saja dilakukan dengan melakukan pemilihan terhadap beberapa objek awal cluster sekaligus dan kemudian melakukan penggabungan objek kedalamnya secara bersamaan

**Optimizing threshold** merupakan pengembangan dari sequential dan pararel dengan melakukan optimalisasi penempatan objek dengan melakukan reassigned kedalam cluster untuk mengoptimalkan suatu kriteria secara menyeluruh, seperti verage within distance untuk sejumlah cluster tertentu.

### **Algoritma K-Means Cluster Analysis**

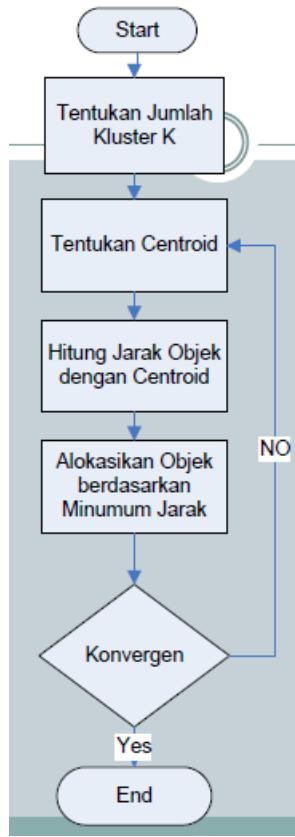
Jika diberikan sekumpulan data  $X=(x_1, x_2, \dots, x_n)$  maka algoritma k-means cluster analysis akan mempartisi  $X$  dalam  $k$  buah cluster, setiap cluster memiliki centroid (titik tengah) atau mean dari data-data dalam cluster tersebut.

Pada tahap awal algoritma k-means cluster analysis akan memilih secara acak  $k$  buah data sebagai centroid (titik tengah), kemudian jarak antara data dengan centroid dihitung dengan menggunakan Euclidean distance, data akan ditempatkan dalam cluster yang terdekat dihitung dari titik tengah cluster.

Centroid baru akan ditetapkan jika semua data sudah ditempatkan dalam cluster terdekat. Proses penentuan centroid dan penempatan data dalam cluster diulangi sampai nilai centroid konvergen (centroid dari semua cluster tidak berubah lagi)

Secara umum K-Means Cluster analysis menggunakan algoritma sebagai berikut:

- 1.Tentukan  $k$  sebagai jumlah cluster yang akan dibentuk
- 2.Bangkitkan  $k$  Centroid (titik pusat cluster) awal secara random
- 3.Hitung jarak setiap data ke masing-masing centroid dari masing-masing cluster
- 4.Alokasikan masing-masing data kedalam centroid yang paling terdekat
- 5.Lakukan iterasi, kemudian tentukan posisi centroid baru dengan cara menghitung rata-rata dari data-data yang berada pada centroid yang sama
- 6.Ulangi langkah 3 jika posisi centroid baru dan centroid lama tidak sama



### Menentukan Banyaknya *Cluster k*

Untuk menentukan nilai banyaknya *cluster k* dilakukan dengan beberapa pertimbangan sebagai berikut :

1. Pertimbangan teoritis, konseptual, praktis yang mungkin diusulkan untuk menentukan berapa banyak jumlah *cluster*.
2. Besarnya relative *cluster* seharusnya bermanfaat, pemecahan *cluster* yang menghasilkan 1 objek anggota *cluster* dikatakan tidak bermanfaat sehingga hal ini perlu untuk dihindari.

### Menentukan *Centroid*

Penentuan *centroid* awal dilakukan secara *random/acak* dari data/objek yang tersedia sebanyak jumlah kluster *k*, kemudian untuk menghitung *centroid cluster* berikutnya ke *i*, *vi* digunakan rumus sebagai berikut :

$$V_k = \frac{\sum_{i=1}^{N_k} X_i}{N_k}$$

- *Vk* : *centroid* pada *cluster* ke *k*
- *Xi* : Data ke *i*
- *Nk* : Banyaknya objek/jumlah data yang menjadi anggota *cluster* ke *k*

### Menghitung Jarak Antara Data Dengan *Centroid*

Untuk menghitung jarak antara data dengan *centroid* terdapat beberapa cara yang dapat dilakukan yaitu *Manhattan/City Block distance (L1)*, *Euclidean Distance (L2)*. Jarak antara dua titik  $X_1$  dan  $X_2$  pada *manhattan/citi block* dihitung dengan menggunakan rumus :

$$D_{L_1}(x_2, x_1) = \|x_2 - x_1\|_1 = \sum_{j=1}^p |x_{2j} - x_{1j}|$$

Dimana P : Dimensi data

| . | : Nilai Absolut

Sedangkan untuk *euclidean distance* jarak antara data dengan *centroid* dihitung dengan menggunakan rumus :

$$D_{L_2}(x_2, x_1) = \|x_2 - x_1\|_2 = \sqrt{\sum_{j=1}^p (x_{2j} - x_{1j})^2}$$

Dimana P : Dimensi data

| . | : Nilai Absolut

**Pengalokasian Ulang Data Kedalam Masing-masing Cluster** Untuk melakukan pengalokasian data kedalam masing-masing *cluster* pada saat iterasi dilakukan secara umum dengan dua cara yaitu dengan cara pengalokasian dengan cara *hard k-means*, dimana secara tegas setiap objek dinyatakan sebagai anggota *cluster* satu dan tidak menjadi anggota *cluster* lainnya. Cara lain adalah dengan cara *fuzzy k-means* dimana masing-masing objek diberikan nilai kemungkinan untuk bisa bergabung dengan setiap *cluster* yang ada. *Hard K-means*, pengalokasian kembali objek kedalam masing-masing *cluster* pada metoda *hard K-means* didasarkan pada perbandingan jarak antara data dengan *centroid* setiap *cluster* yang ada, objek dialokasikan secara tegas kedalam *cluster* yang mempunyai jarak ke *centroid* terdekat dengan data tersebut. Pengalokasian ini dirumuskan sebagai berikut :

$$a_{ik} = \begin{cases} 1 & d = \min\{D(x_k, v_i)\} \\ 0 & \text{lainnya} \end{cases}$$

aik : keanggotaan data atau objek ke  $k$  pada  $cluster$  ke  $i$

vi : Nilai  $centroid cluster$  ke  $i$

*fuzzy k-means*, pada *fuzzy k-means* atau lebih sering disebut *fuzzy c-means* mengalokasikan kembali objek atau data kedalam masing-masing  $cluster$  dengan menggunakan *membership function*,  $u_{ik}$ , yang merujuk pada seberapa besar suatu objek atau data bisa menjadi anggota suatu  $cluster$ .

Pada *fuzzy k-means* yang diusulkan oleh Bezdek diperkenalkan juga suatu variable  $m$  yang merupakan *weighting exponent* dari *membership function*.  $m$  mempunyai wilayah nilai  $m > 1$ , sampai sekarang belum jelas berapa nilai  $m$  yang optimal dalam melakukan proses optimalisasi suatu permasalahan *clustering*.

Nilai  $m$  yang umum digunakan adalah 2. *Membership function* untuk suatu data kedalam suatu  $cluster$  tertentu dihitung dengan menggunakan rumus :

$$u_{ik} = \sum_{j=1}^c \left( \frac{D(x_k, v_i)}{D(x_k, v_j)} \right)^{\frac{2}{m-1}}$$

Dimana

$u_{ik}$  : membership function untuk data atau objek ke  $k$  pada  $cluster$  ke  $i$

$v_i$  : Nilai  $centroid cluster$  ke  $i$

$m$  : Weighting component

c : Jumlah  $cluster$

### **Konvergensi**

Pengecekan *konvergensi* dilakukan dengan membandingkan matrik *group assignment* pada iterasi sebelumnya dengan matrik *group assignment* pada iterasi yang sedang berjalan.

Jika hasilnya sama maka *algoritma k-means cluster analysis* sudah *konvergen*, tetapi jika berbeda maka belum *konvergen* sehingga perlu dilakukan iterasi berikutnya

### **Menilai Kualitas Cluster**

Hasil dari *cluster analysis* yang bagus jika setiap  $cluster$  memiliki tingkat similaritas yang tinggi satu sama lain (*internal homogeneity*) diukur dengan variance dalam  $cluster$  Vw yang

sama sekali berbeda dengan nilai anggota *cluster* yang lain (*external homogeneity*) yang diukur dengan varian antar *cluster* Vb

*Cluster* dianggap ideal jika mempunya Vw seminimal mungkin dan Vb semaksimal mungkin, sehingga nilai *homogeneity* dapat dirumuskan sebagai berikut :

$$V_{Min} = \frac{V_w}{V_b}$$

untuk rumus ini maka semakin kecil nilai *Vmin* maka *homogeneity* semakin bagus, atau *homogeneity* juga dapat dirumuskan sebagai berikut :

$$V_{Max} = \frac{V_b}{V_w}$$

untuk rumus ini maka semakin besar nilai *Vmax* maka *homogeneity* semakin bagus

Untuk menghitung nilai varians dari semua data pada tiap *cluster* dapat dilakukan dengan menggunakan rumus :

$$V_c^2 = \frac{1}{n_c - 1} \sum_{i=1}^{n_c} \left( d_i - \bar{d}_i \right)^2$$

Dimana

$V_c^2$  = variance pada *cluster* c

c = 1..k dimana k = jumlah *cluster*

$n_c$  = Jumlah data pada *cluster* ke c

$d_i$  = data ke- i pada suatu *cluster*

$\bar{d}_i$  = rata-rata atau *centroid* dari data pada suatu *cluster* id

Sedangkan menghitung variance dalam *cluster* dapat dihitung dengan menggunakan rumus :

$$V_w = \frac{1}{N-k} \sum_{i=1}^k (n_i - 1) \cdot V_i^2$$

Dimana  $Vw$  = Varians dalam *cluster*

N = Jumlah semua data

k = Banyaknya *cluster*

$n_i$  = Jumlah data dalam *cluster* ke i

$v_{i2}$  = Variance pada *cluster* ke i

Sedangkan untuk menghitung varians antar *cluster* dihitung dengan menggunakan rumus :

$$v_b = \frac{1}{k-1} \sum_{i=1}^k n_i (\bar{d}_i - \bar{d})^2$$

Dimana

$$\bar{d} = \text{rata-rata } \bar{d}_i$$

Sedangkan nilai variance dari semua *cluster* diperoleh dengan membagi nilai variance dalam *cluster* dengan nilai variance antar *cluster*, dimana semakin kecil nilai tersebut maka semakin bagus *cluster* yang dihasilkan.

### Beberapa Permasalahan *K-Means Cluster Analysis*

Terdapat beberapa permasalahan yang sering ditemukan pada pemakaian algoritma *K-means Cluster Analysis*, antara lain yaitu :

1. Pemilihan jumlah custer yang tepat
2. Ditemukannya beberapa hasil *cluster* yang berbeda.
3. Nilai distance yang sama, sehingga berpengaruh pada alokasi data dalam *cluster*
4. Kegagalan *Konvergensi*
5. Pendekslan Outlier

**Contoh Penerapan Algoritma *K-Means Cluster Analysis*** Misalkan kita mempunyai dua variable X<sub>1</sub> dan X<sub>2</sub> dengan masing-masing mempunyai item-item A, B, C dan D sebagai berikut :

Item	Observasi	
	X <sub>1</sub>	X <sub>2</sub>
A	1	1
B	2	1
C	4	3
D	5	4

Tujuannya adalah membagi semua item menjadi 2 *cluster* ( $k = 2$ ) , dengan menggunakan algoritma yang disebutkan diatas maka langkah-langkah yang dikerjakan adalah sebagai berikut :

- 1.Tentukan k sebagai jumlah *cluster* yang akan di bentuk  $k = 2$
- 2.Bangkitkan  $k$  *Centroid* (titik pusat *cluster*) awal secara *random* Secara *random* kita tentukan A dan B sebagai *centroid* yang pertama, sehingga diperoleh  $c_1=(1,1)$  dan  $c_2=(2,1)$
- 3.Hitung jarak setiap data ke masing-masing *centroid* dari masing-masing *cluster* dengan *Euclidian distance* sebagai berikut :

$$D_{L_2}(x_2, x_1) = \|x_2 - x_1\|_2 = \sqrt{\sum_{j=1}^p (x_{2j} - x_{1j})^2}$$

Dimana

P : Dimensi data

| . | : Nilai Absolut

$D(C_1, B) =$	$\sqrt{(2-1)^2 + (1-1)^2} = 1$
$D(C_1, C) =$	$\sqrt{(4-1)^2 + (3-1)^2} = 3,61$
$D(C_1, D) =$	$\sqrt{(5-1)^2 + (4-1)^2} = 5$
$D(C_2, A) =$	$\sqrt{(1-2)^2 + (1-1)^2} = 1$
$D(C_2, B) =$	$\sqrt{(2-2)^2 + (1-1)^2} = 0$
$D(C_2, C) =$	$\sqrt{(4-2)^2 + (3-1)^2} = 2,83$
$D(C_2, D) =$	$\sqrt{(5-2)^2 + (4-1)^2} = 4,24$

Sehingga *distance* yang diperoleh adalah sebagai berikut :

Cluster Centroid	Distance			
	A	B	C	D
<b>C<sub>1</sub></b>	0	1	3,61	5
<b>C<sub>2</sub></b>	1	0	2,83	4,24

4. Alokasikan masing-masing data ke dalam *centroid* yang paling terdekat
  - Proses alokasi dilakukan dengan melihat minimum distance.
  - Dari table distance diatas maka terlihat bahwa jarak item A terdekat pada *cluster* C1 sehingga item A dialokasikan kepada *cluster* C1,

- Sementara item B, Item C, Item D jarak terdekatnya pada *cluster* C2, sehingga item B, C, D dialokasikan pada *cluster* C2.
  - Dengan menggunakan rumus alokasi dibawah ini,
- $$a_{ik} = \begin{cases} 1 & d = \min\{D(x_k, v_i)\} \\ 0 & \text{lainnya} \end{cases}$$
- Maka diperoleh *table group assignmentnya* adalah sebagai berikut :

- |          | <b>A</b> | <b>B</b> | <b>C</b> | <b>D</b> |
|----------|----------|----------|----------|----------|
| <b>1</b> | o        | o        | o        |          |
| <b>0</b> | 1        | 1        | 1        |          |

5. Lakukan iterasi-1, kemudian tentukan posisi *centroid* baru dengan cara menghitung rata-rata dari data-data yang berada pada *centroid* yang sama. Dengan menggunakan rumus,

$$V_i = \frac{\sum_{k=1}^{N_i} X_k}{N_i}$$

- Maka diperoleh *centroid* baru untuk kedua *cluster* tersebut adalah

$$C_1 = (1,1), \text{ karena beranggotakan 1 anggota}$$

$$C_{2(x_1)} = \frac{2+4+5}{3} = 3,67$$

$$C_{2(x_2)} = \frac{1+3+4}{3} = 2,67$$

$$C_2 = (3,67, 2,67)$$

6. Ulangi langkah 3 jika posisi *centroid* baru dan *centroid* lama tidak sama, karena nilai *centroid*nya berbeda maka langkah no 3 diulangi kembali sebagai berikut :

- $D^1(C_1, A) = \sqrt{(1-1)^2 + (1-1)^2} = 0$
- $D^1(C_1, B) = \sqrt{(2-1)^2 + (1-1)^2} = 1$
- $D^1(C_1, C) = \sqrt{(4-1)^2 + (3-1)^2} = 3,61$
- $D^1(C_1, D) = \sqrt{(5-1)^2 + (4-1)^2} = 5$
- $D^1(C_2, A) = \sqrt{(1-3,67)^2 + (1-2,67)^2} = 3,14$
- $D^1(C_2, B) = \sqrt{(2-3,67)^2 + (1-2,67)^2} = 2,36$
- $D^1(C_2, C) = \sqrt{(4-3,67)^2 + (3-2,67)^2} = 0,47$
- $D^1(C_2, D) = \sqrt{(5-3,67)^2 + (4-2,67)^2} = 1,89$

- Sehingga *distance* yang diperoleh pada iterasi 1 adalah sebagai berikut

Cluster Centroid	Distance			
	A	B	C	D
$C_1$	0	1	3,61	5
$C_2$	3,14	2,36	0,47	1,89

- Alokasikan masing-masing data ke dalam *centroid* yang paling terdekat Maka diperoleh *table group assignment*nya pada iterasi 1 adalah sebagai berikut :

A	B	C	D
1	1	0	0
0	0	1	1

- Karena hasil *table group assignment* pada iterasi 1 berbeda dengan *table group assignment* sebelumnya maka hasilnya belum konvergen sehingga perlu dilakukan iterasi berikutnya, sebagai berikut
- Lakukan iterasi-2, tentukan posisi *centroid* baru dengan cara menghitung rata-rata dari data-data yang berada pada *centroid* yang sama.
- Maka diperoleh *centroid* baru untuk kedua *cluster* tersebut adalah

$C_{1(x_1)} = \frac{1+2}{2} = 1,5$
$C_{1(x_2)} = \frac{1+1}{2} = 1$
• $C_1 = (1,5, 1)$
$C_{2(x_1)} = \frac{4+5}{2} = 4,5$
$C_{2(x_2)} = \frac{3+4}{2} = 3,5$
• $C_2 = (4,5, 3,5)$

- Karena nilai *centroid*-nya berbeda dengan iterasi 1 maka langkah berikutnya menghitung kembali *distance*-nya sebagai berikut :

• $D^2(C_1, A) = \sqrt{(1-1,5)^2 + (1-1)^2} = 0,5$
• $D^2(C_1, B) = \sqrt{(2-1,5)^2 + (1-1)^2} = 0,5$
• $D^2(C_1, C) = \sqrt{(4-1,5)^2 + (3-1)^2} = 3,2$
• $D^2(C_1, D) = \sqrt{(5-1,5)^2 + (4-1)^2} = 4,61$
• $D^2(C_2, A) = \sqrt{(1-4,5)^2 + (1-3,5)^2} = 4,30$
• $D^2(C_2, B) = \sqrt{(2-4,5)^2 + (1-3,5)^2} = 3,54$
• $D^2(C_2, C) = \sqrt{(4-4,5)^2 + (3-3,5)^2} = 0,71$
• $D^2(C_2, D) = \sqrt{(5-4,5)^2 + (4-3,5)^2} = 0,71$

- Sehingga *distance* yang diperoleh pada iterasi 1 adalah sebagai berikut :

Cluster Centroid	Distance			
	A	B	C	D
<b>C<sub>1</sub></b>	0,5	0,5	3,2	4,61
<b>C<sub>2</sub></b>	4,3	3,54	0,71	0,71

- Alokasikan masing-masing data ke dalam *centroid* yang paling terdekat
- Maka diperoleh *table group assignmentnya* pada iterasi 2 adalah sebagai berikut :

A	B	C	D
<b>1</b>	<b>1</b>	0	0
0	0	1	1

- Dari hasil *table assignment* pada iterasi 2 ternyata hasilnya sama dengan *table group assignment* pada iterasi 1 sehingga pada iterasi 2 ini *sudah konvergen* sehingga tidak perlu dilakukan iterasi kembali, dan hasil akhir *cluster* yg diperoleh adalah :

Item	Observasi		Cluster
	X <sub>1</sub>	X <sub>2</sub>	
A	1	1	1
B	2	1	1
C	4	3	2
D	5	4	2

### Sumber Ringkasan :

<http://eprints.binadarma.ac.id/529/1/DATA%20MINING%20%26%20WAREHOUSE%20materi%202.pdf>

<https://mulaab.github.io/datamining/memahami-data/>

[https://www.youtube.com/watch?v=QOJ8Fiy\\_W4Y](https://www.youtube.com/watch?v=QOJ8Fiy_W4Y)

Han, J., Kamber, M., Pei, J.: Data Mining Concept and Techniques, 3rd ed. Morgan Kaufmann-Elsevier, Amsterdam (2012)

Jain. A.K (2009). Data Clustering: 50 Years Beyond K-Means. Pattern Recognition Letters, 2009.

Tan, P.N., Steinbach, M., Kumar, V. (2006) *Introduction to Data Mining*. Boston:Pearson Education.

Nama : Oman Arrohman

Mata Kuliah : Advanced Database

NIM : 202420042

Berikut ini adalah tahap-tahap pengukuran jarak antar data dengan algoritma K-Means clustering :

1. Input data yang digunakan dalam clustering. Data ini digunakan untuk menentukan nilai rata-rata data yang berada dalam satu cluster dan menentukan jarak dari setiap data ke centroid.
2. Alokasi ke Cluster secara acak, dalam tahap ini pertama kalinya data dialokasikan ke cluster secara acak tanpa ada kriteria tertentu.
3. Hitung centroid data yang ada pada setiap cluster. Nilai centroid pada k-means digunakan sebagai pusat cluster. Dengan menentukan anggota cluster secara acak pada tahap sebelumnya, maka terbentuk iterasi awal sebagai pusat cluster acak.
4. Alokasi ke centroid terdekat, pada tahap ini hasil centroid dari setiap cluster sudah diketahui, kemudian data dialokasikan pada centroid terdekat berdasarkan nilai jarak similarity data terhadap centroid. Jarak similarity dari data ke centroid pada masing-masing cluster diperoleh dari perhitungan Euclidean Distance. Kemudian nilai jarak setiap data ke centroid cluster dibandingkan, dan data point menjadi anggota dari cluster berdasarkan jarak data ke centroid terdekat.
5. Konvergen, mengalokasikan data point ke centroid dengan nilai jarak terdekat, dengan menguji apakah cluster yang terbentuk telah membentuk cluster yang konvergen atau tidak. Cluster dinyatakan konvergen jika anggota dari masing-masing cluster yang terbentuk tidak mengalami perubahan anggota. Tetapi jika masih terjadi perubahan anggota cluster, maka akan kembali dilakukan tahapan menghitung centroid dari masing-masing cluster yang terbentuk dan diikuti dengan perhitungan nilai similarity ke centroid yang baru terbentuk. Proses tersebut terus berulang sampai hasil cluster konvergen.

Sumber :

<https://ejurnal.gunadarma.ac.id/index.php/tekno/article/view/1599/1358>

[https://www.youtube.com/watch?v=8mXL2z3lg\\_o&t=11s](https://www.youtube.com/watch?v=8mXL2z3lg_o&t=11s)

## Tugas 01

Nama : Puspita Dewi Setyadi

NIM : 202420011

Setiap pengukuran jarak antar data dapat diimplementasikan sesuai apa yang akan diukur. Pada sebuah variabel yang memerlukan skala tertentu misal tentang jenis kelamin sudah pasti skala nominal yang kita pakai pilihannya hanya dua yaitu laki-laki dan perempuan, lalu ada metode analisis yang kita pakai membutuhkan persyaratan data tertentu misal membutuhkan data nominal, dependen pada analisis diskriminan. Apabila pengujian dilakukan dengan alat ini maka skala yang menghasilkan data nominal yaitu skala dikotomi. Sedangkan Data nominal dan data ordinal yaitu data kategoris yang digunakan jika melakukan *corresponce analysis*. Skala yang sesuai adalah dikotomi dan *itemized-rating scale*.

Sumber link : <https://www.bilsonsimamora.com/skala-pengukuran/?print=print>

**TUGAS 01**  
**Tutorial Jarak Data**



Dibuat Oleh

**Robby Prabowo**

**202420001**

**MTIA1**

Dosen Pengampu

**TRI BASUKI KURNIAWAN, S.Kom, M.Eng, Ph.D**

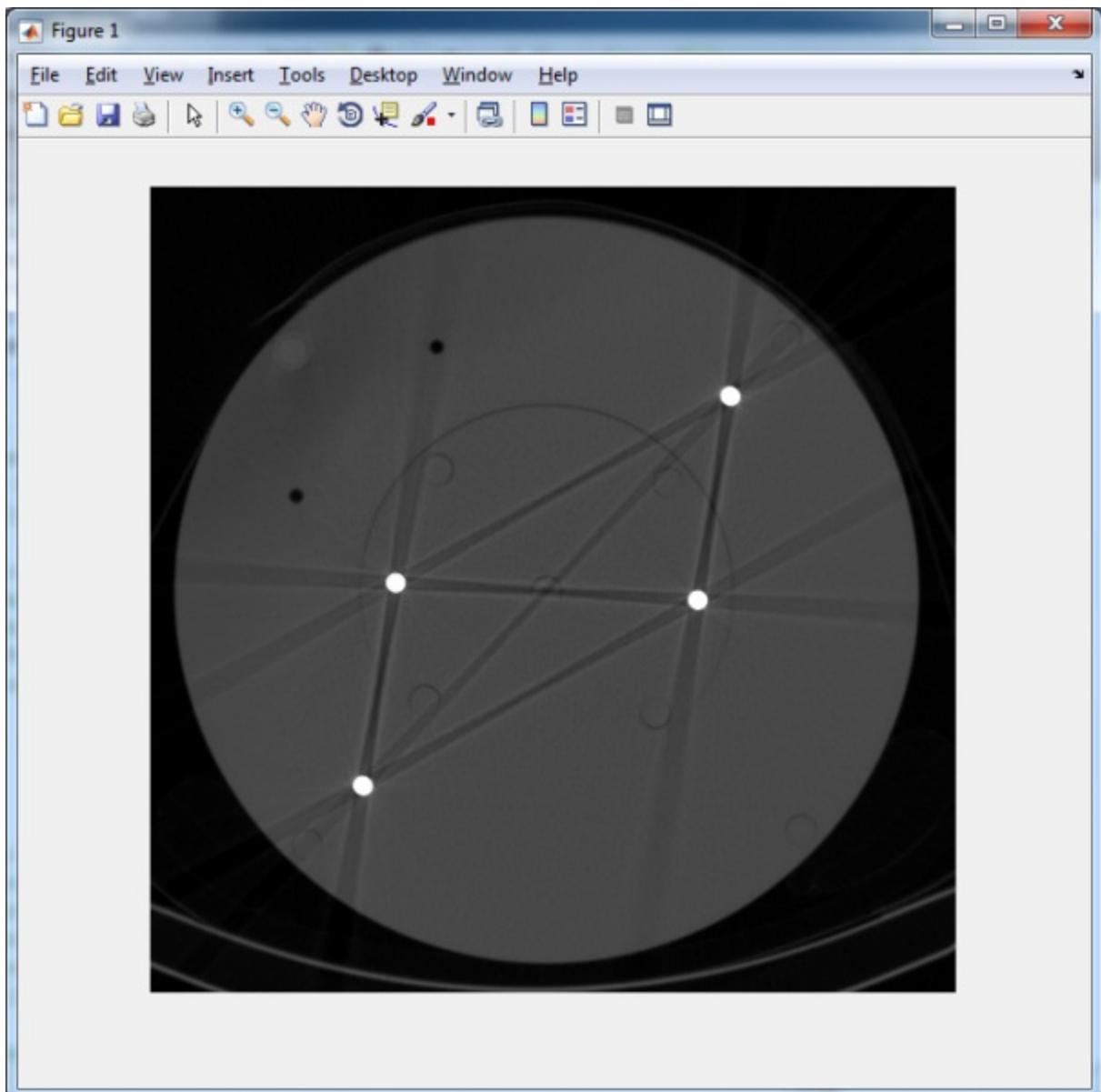
**Program Pasca Sarjana**

**Universitas Binadarma Palembang**

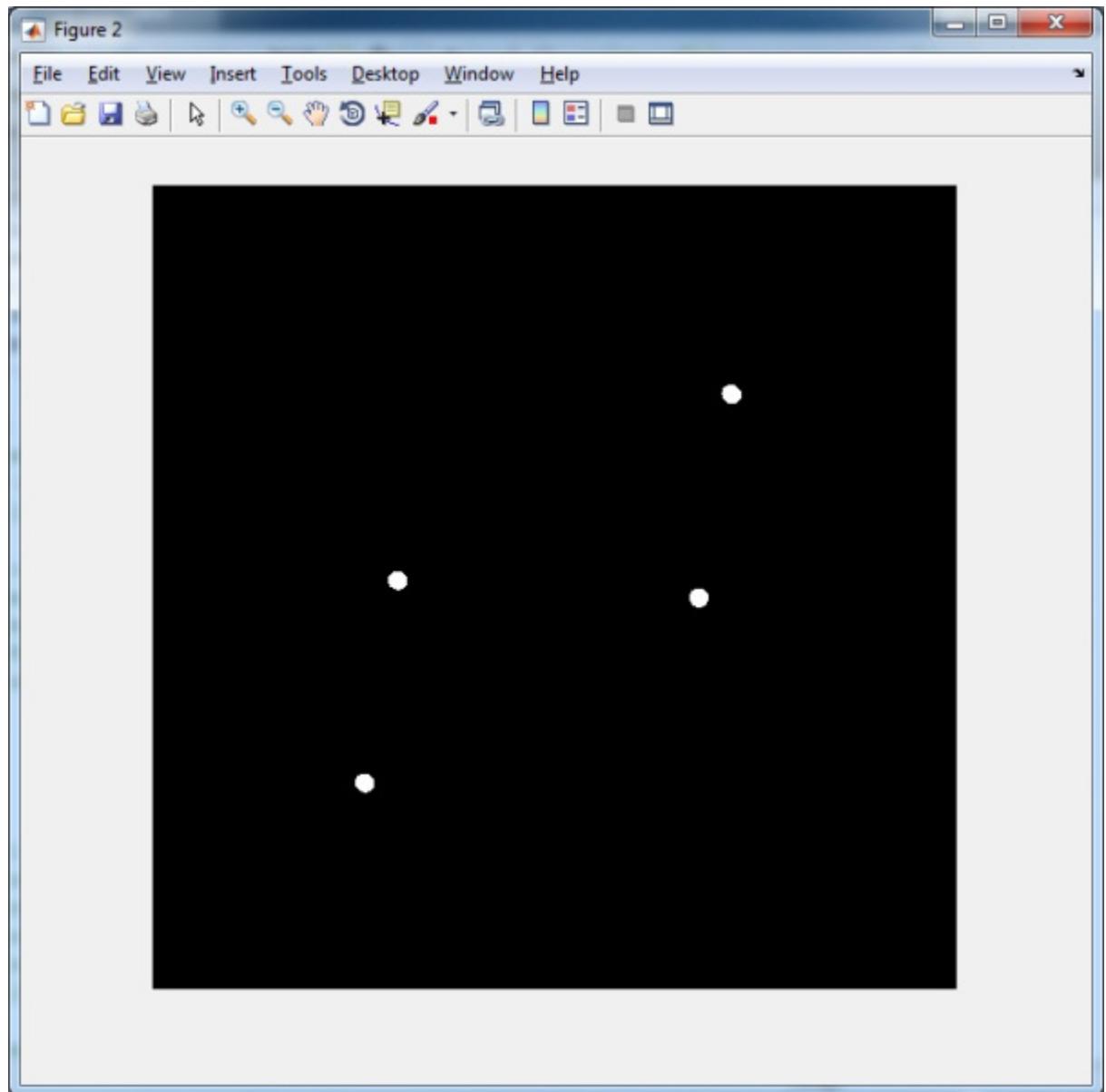
**2020/2021**

Dalam sistem koordinat citra dua dimensi, jarak antara dua objek dapat diukur menggunakan persamaan [euclidean distance](#). Berikut ini merupakan contoh aplikasi pemrograman matlab untuk mengukur jarak antara dua objek dalam citra phantom berekstensi dicom. Langkah-langkahnya adalah sebagai berikut:

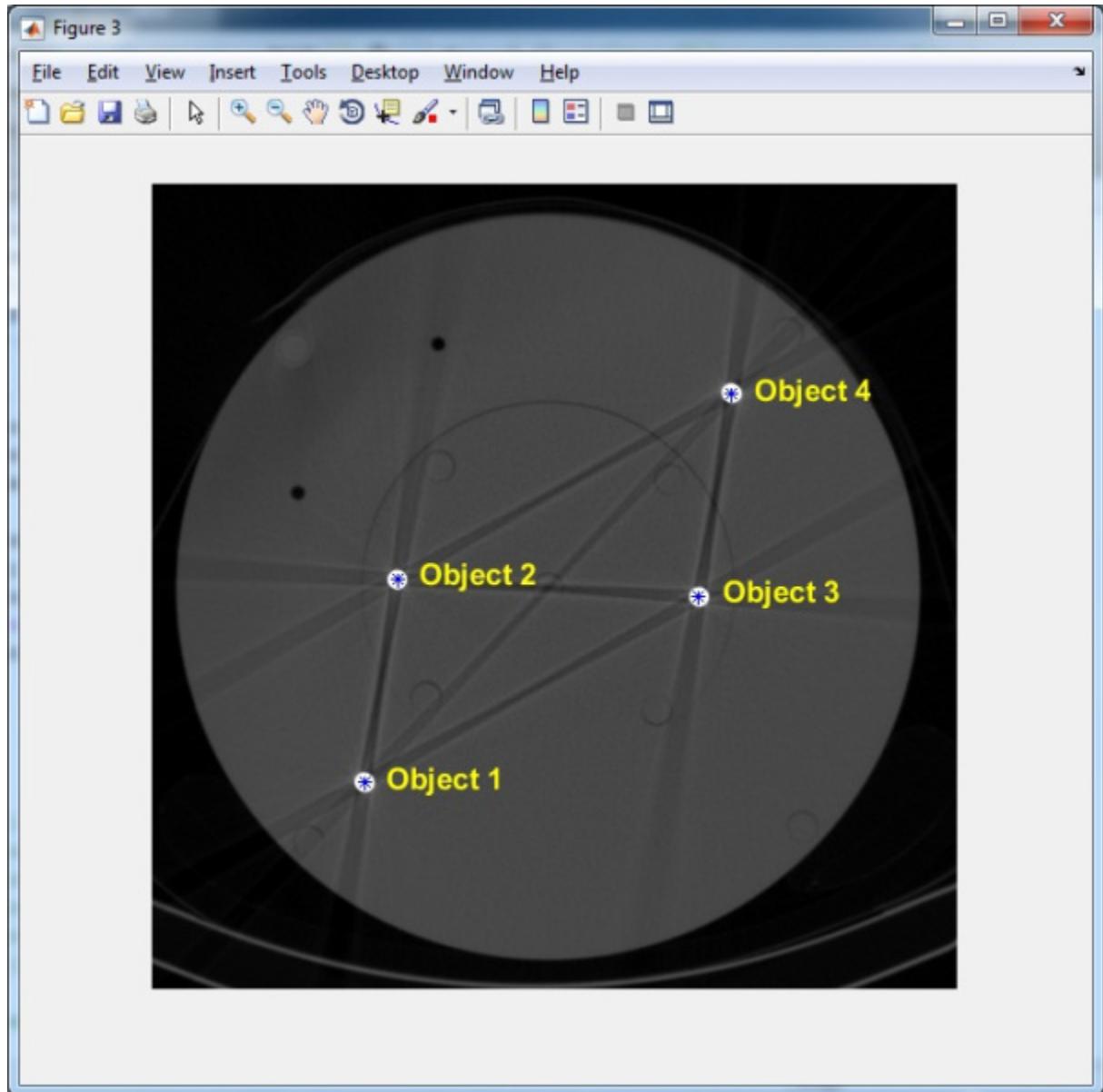
1. Membaca citra phantom yang berekstensi dicom



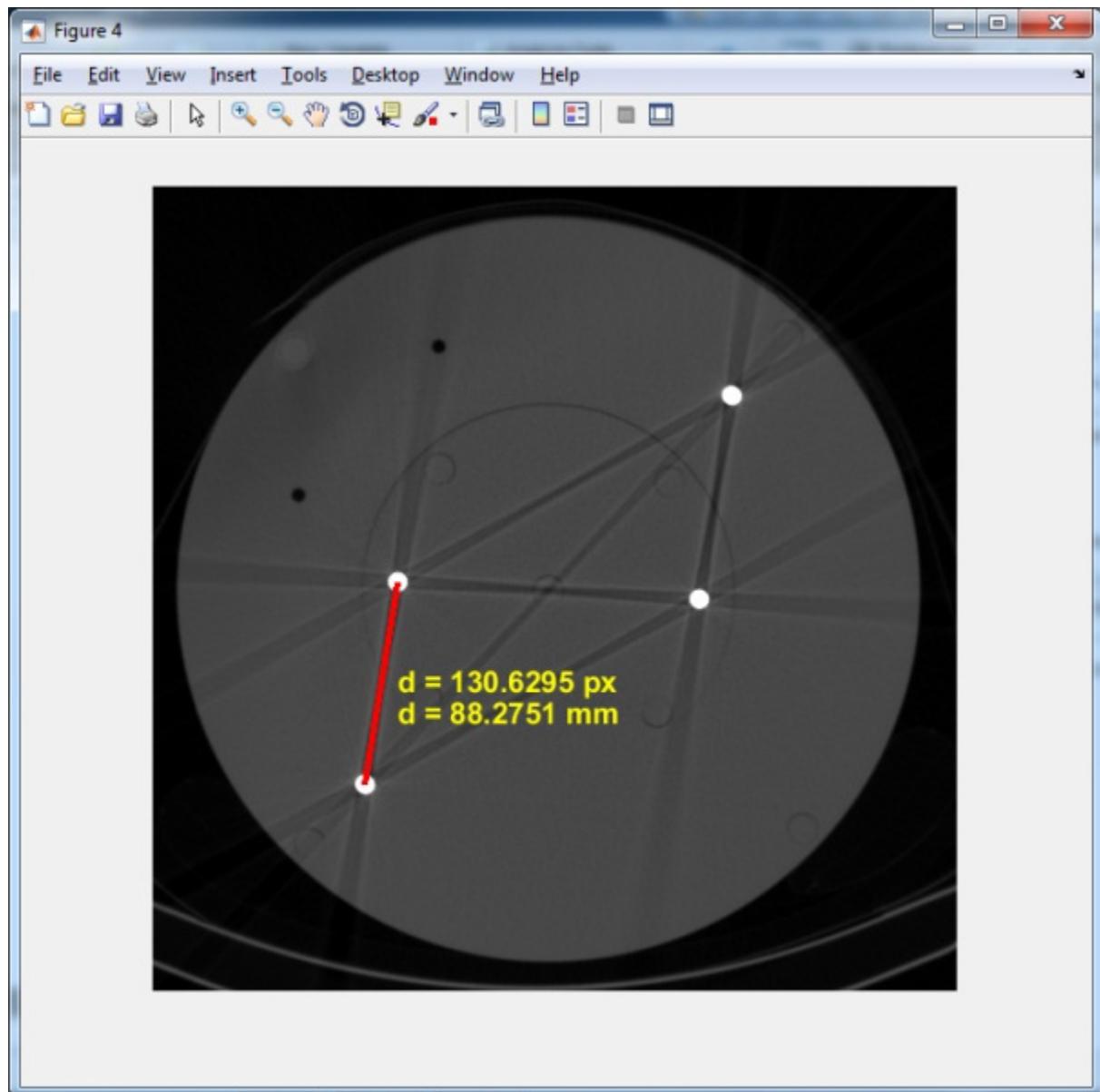
2. Melakukan operasi thresholding citra



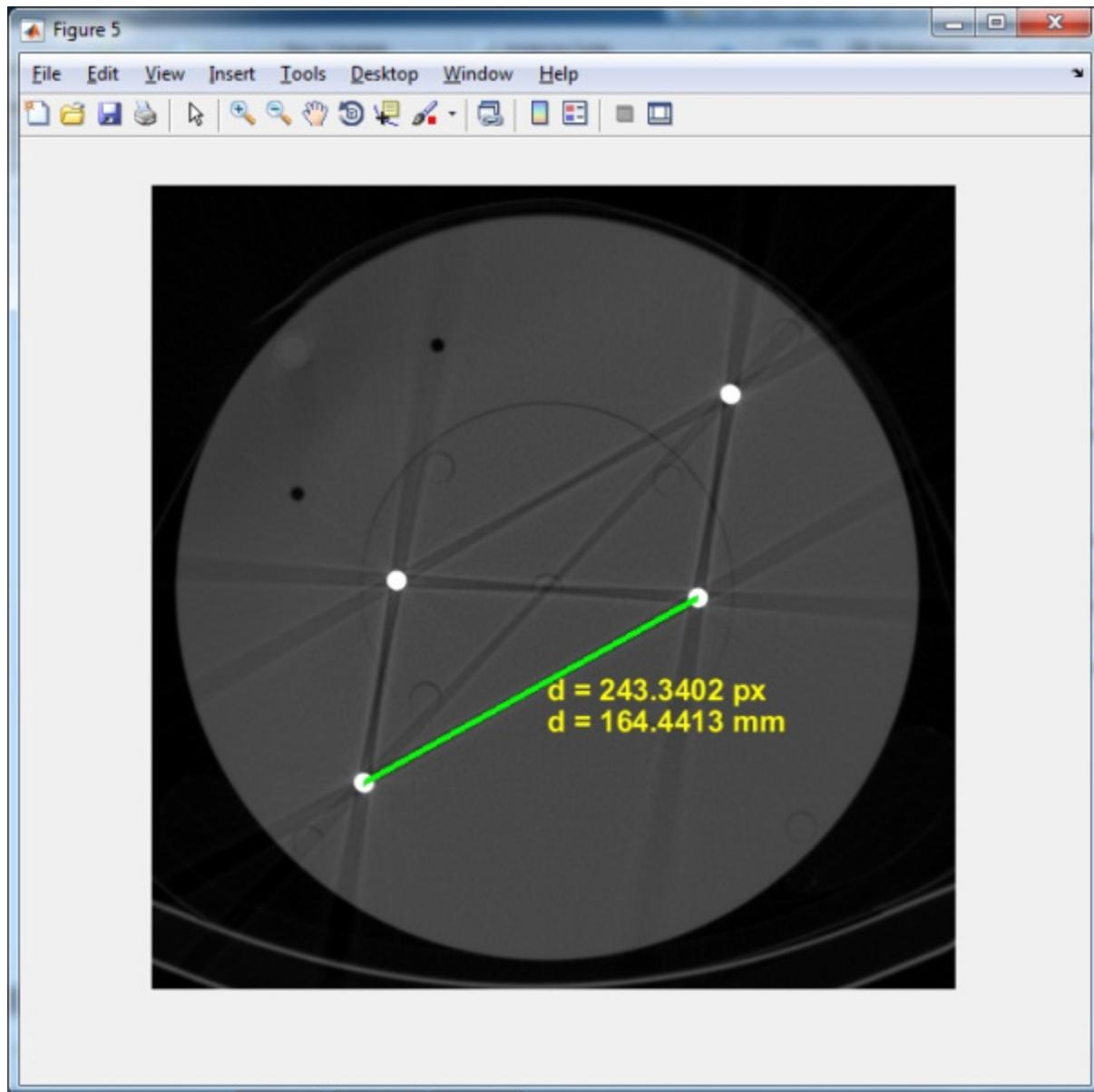
3. Menentukan centroid dan labelling objek



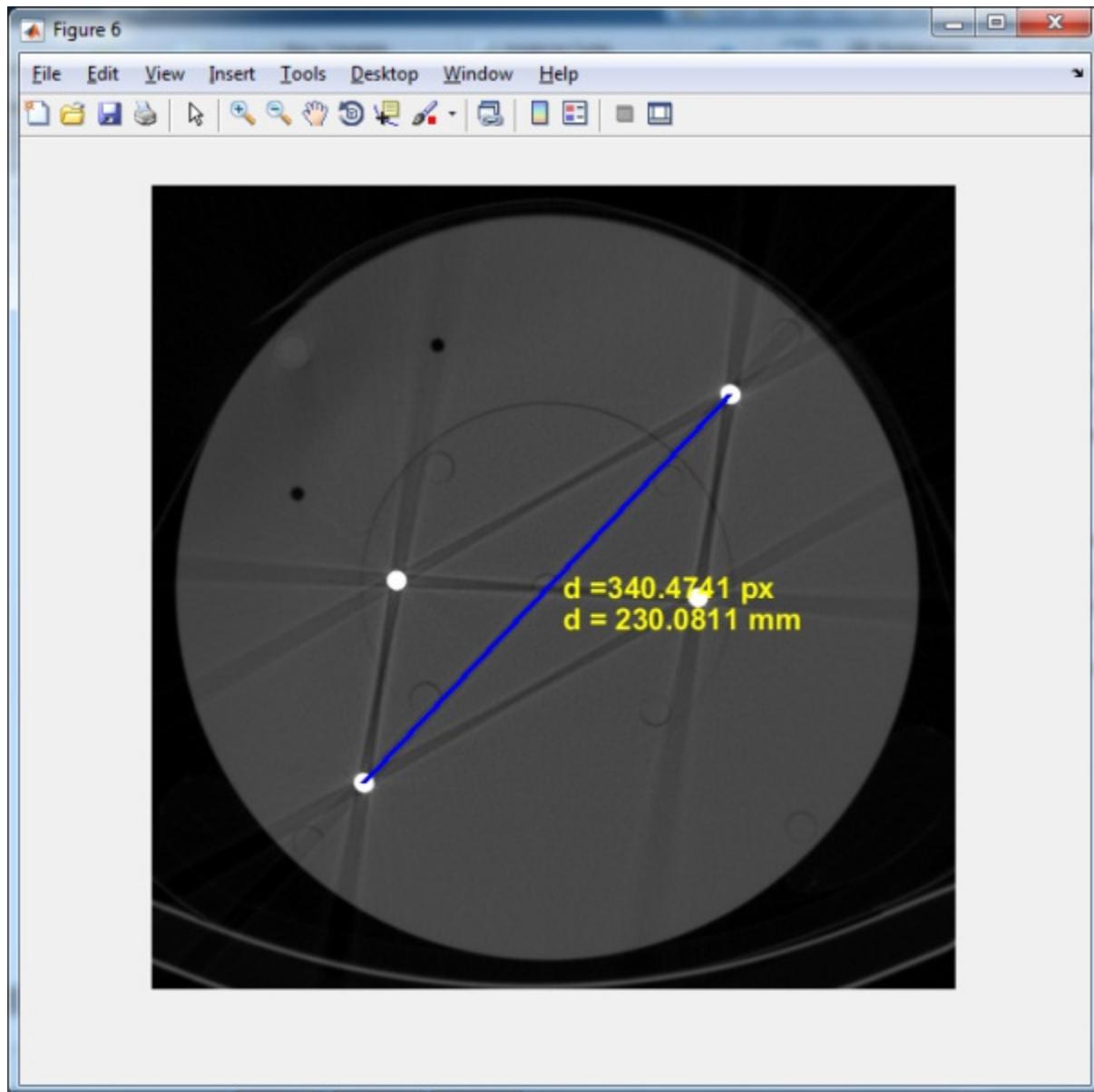
4. Mengukur jarak antara dua objek menggunakan persamaan euclidean distance dalam satuan piksel. Satuan piksel kemudian dikonversi menjadi satuan mm dengan cara membagi hasil pengukuran jarak dalam satuan piksel dengan resolusi spasial (pada contoh ini diketahui resolusi spasial citra dicom adalah sebesar 1,4798 piksel per mm). Pengukuran jarak antara objek 1 dengan objek 2 ditunjukkan pada gambar berikut:



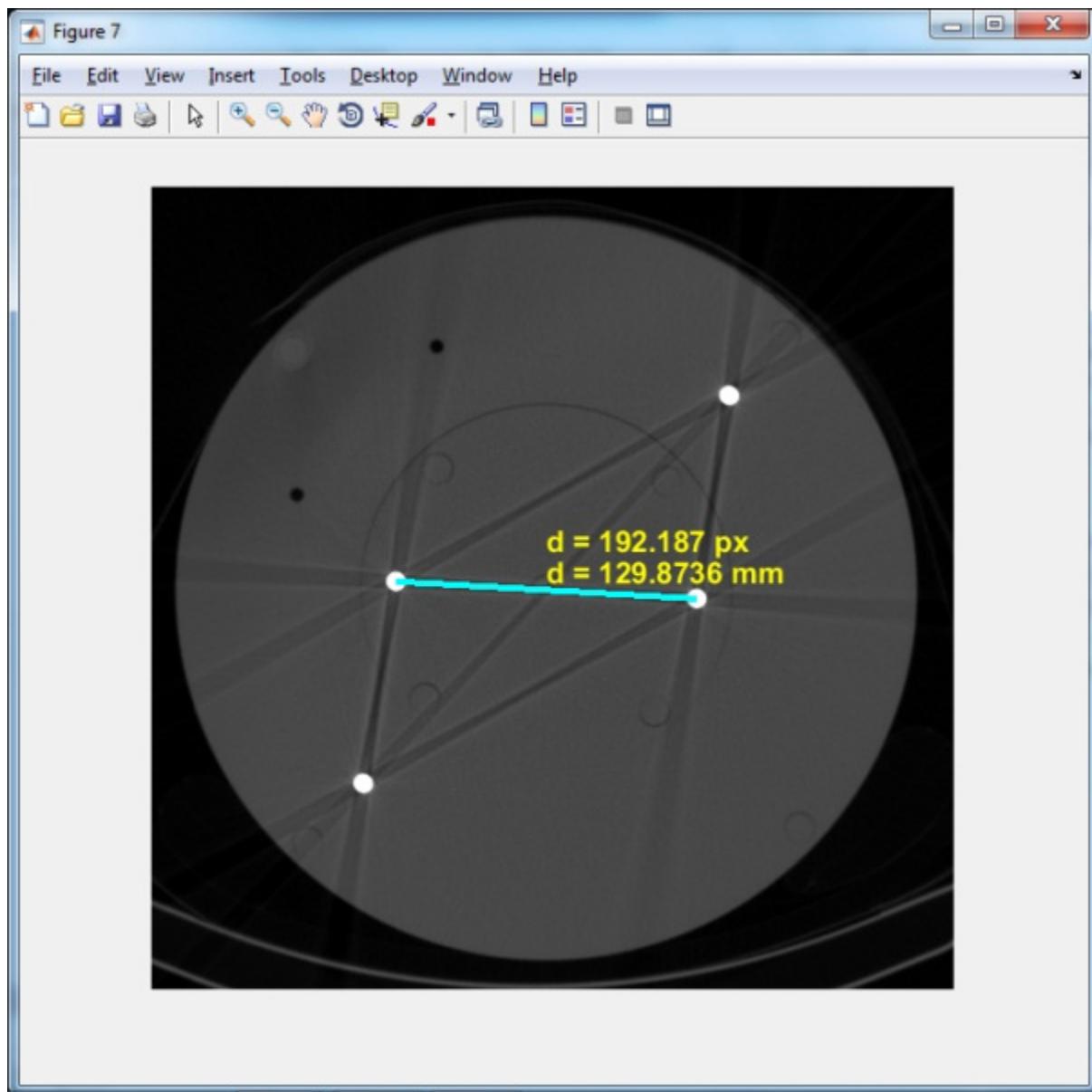
5. Pengukuran jarak antara objek 1 dan objek 3



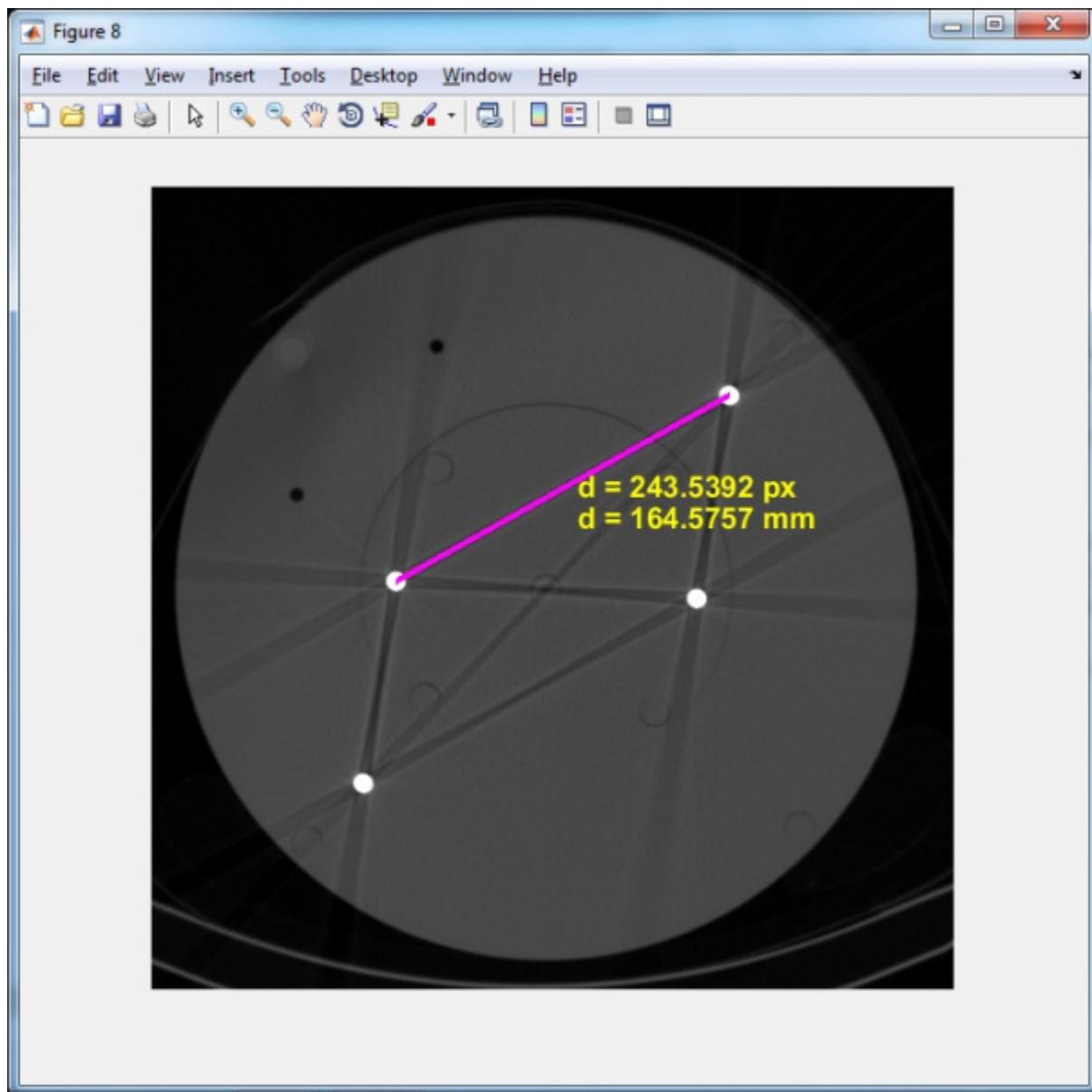
6. Pengukuran jarak antara objek 1 dan objek 4



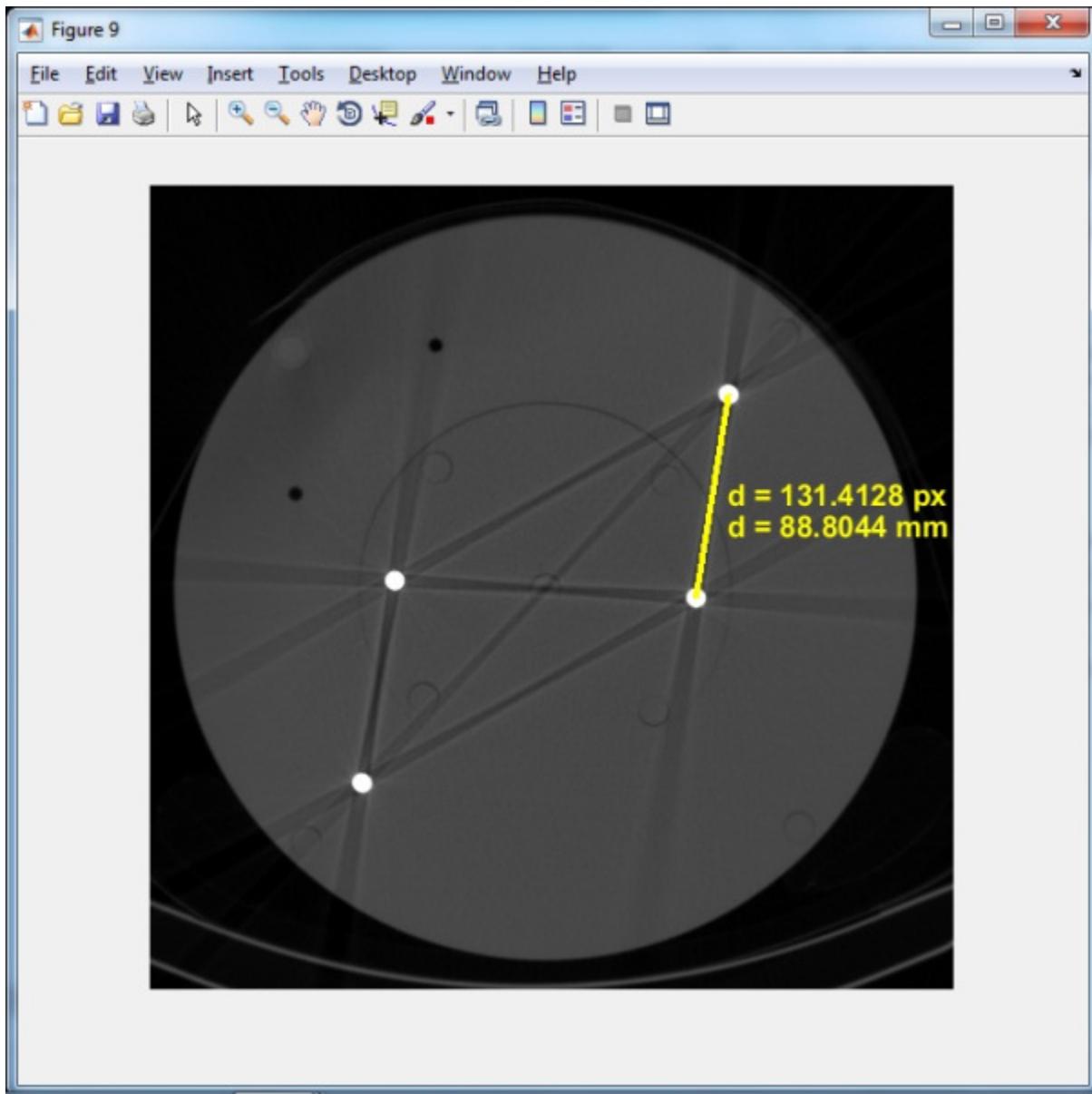
7. Pengukuran jarak antara objek 2 dan objek 3



8. Pengukuran jarak antara objek 2 dan objek 4



9. Pengukuran jarak antara objek 3 dan objek 4



Tampilan source code Matlab untuk mengukur jarak antara dua objek dalam citra adalah sebagai berikut:

```
1 clc;clear;close all;
2 I = dicomread('slice1.dcm');
3 figure(1), imshow(I, []);
4
5 BW = I>3000;
6 figure(2), imshow(BW, []);
7
8 s = regionprops(BW, 'centroid');
9 centroids = cat(1,s.Centroid);
```

```

10
11 % labelling
12 [B,L] = bwboundaries(BW,'noholes');
13 [~,num] = bwlabel(BW,8);
14 figure(3), imshow(I,[])
15 hold on
16 for k = 1:num
17     boundary = B{k};
18     text(boundary(1,2)+20,boundary(1,1),strcat(['Object
19 ',num2str(k)]),'Color','y',...
20         'FontSize',14,'FontWeight','bold'));
21     plot(centroids(:,1), centroids(:,2), 'b*')
22 end
23 hold off
24
25 % object 1 & 2
26 x1 = centroids(1,1);
27 y1 = centroids(1,2);
28 x2 = centroids(2,1);
29 y2 = centroids(2,2);
30 figure(4), imshow(I,[])
31 hold on
32 plot([x1;x2], [y1;y2], 'r','LineWidth',3)
33
34 d_px = sum(([x1;y1]-[x2;y2]).^2).^0.5;
35 res = 1.4798;
36 d_mm = d_px/res;
37 text((x1+x2+20)/2,(y1+y2)/2,strcat(['d = ',num2str(d_px),' px']),'Color','y',...
38     'FontSize',14,'FontWeight','bold');
39 text((x1+x2+20)/2,(y1+y2+40)/2,strcat(['d = ',num2str(d_mm),' 
40 mm']),'Color','y',...
        'FontSize',14,'FontWeight','bold');
41 hold off
42
43 % object 1 & 3
44 x1 = centroids(1,1);
45 y1 = centroids(1,2);
46 x3 = centroids(3,1);
47 y3 = centroids(3,2);
48 figure(5), imshow(I,[])
49 hold on
50 plot([x1;x3], [y1;y3], 'g','LineWidth',3)
51
52 d_px = sum(([x1;y1]-[x3;y3]).^2).^0.5;
53 res = 1.4798;
54 d_mm = d_px/res;
55 text((x1+x3+20)/2,(y1+y3)/2,strcat(['d = ',num2str(d_px),' px']),'Color','y',...
56     'FontSize',14,'FontWeight','bold');
57 text((x1+x3+20)/2,(y1+y3+40)/2,strcat(['d = ',num2str(d_mm),' 
58 mm']),'Color','y',...
        'FontSize',14,'FontWeight','bold');
59 hold off
60
61 % object 1 & 4
62 x1 = centroids(1,1);
63 y1 = centroids(1,2);
64 x4 = centroids(4,1);

```

```

66 y4 = centroids(4,2);
67 figure(6), imshow(I,[])
68 hold on
69 plot([x1;x4], [y1;y4], 'b','LineWidth',3)
70
71 d_px = sum(([x1;y1]-[x4;y4]).^2).^0.5;
72 res = 1.4798;
73 d_mm = d_px/res;
74 text((x1+x4+20)/2,(y1+y4)/2,strcat(['d = ',num2str(d_px),' px']),'Color','y',...
75 'FontSize',14,'FontWeight','bold');
76 text((x1+x4+20)/2,(y1+y4+40)/2,strcat(['d = ',num2str(d_mm),' mm']),'Color','y',...
77 'FontSize',14,'FontWeight','bold');
78 hold off
79
80 % object 2 & 3
81 x2 = centroids(2,1);
82 y2 = centroids(2,2);
83 x3 = centroids(3,1);
84 y3 = centroids(3,2);
85 figure(7), imshow(I,[])
86 hold on
87 plot([x2;x3], [y2;y3], 'c','LineWidth',3)
88
89 d_px = sum(([x2;y2]-[x3;y3]).^2).^0.5;
90 res = 1.4798;
91 d_mm = d_px/res;
92 text((x2+x3)/2,(y2+y3-60)/2,strcat(['d = ',num2str(d_px),' px']),'Color','y',...
93 'FontSize',14,'FontWeight','bold');
94 text((x2+x3)/2,(y2+y3-20)/2,strcat(['d = ',num2str(d_mm),' mm']),'Color','y',...
95 'FontSize',14,'FontWeight','bold');
96 hold off
97
98 % object 2 & 4
99 x2 = centroids(2,1);
100 y2 = centroids(2,2);
101 x4 = centroids(4,1);
102 y4 = centroids(4,2);
103 figure(8), imshow(I,[])
104 hold on
105 plot([x2;x4], [y2;y4], 'm','LineWidth',3)
106
107 d_px = sum(([x2;y2]-[x4;y4]).^2).^0.5;
108 res = 1.4798;
109 d_mm = d_px/res;
110 text((x2+x4+20)/2,(y2+y4)/2,strcat(['d = ',num2str(d_px),' px']),'Color','y',...
111 'FontSize',14,'FontWeight','bold');
112 text((x2+x4+20)/2,(y2+y4+40)/2,strcat(['d = ',num2str(d_mm),' mm']),'Color','y',...
113 'FontSize',14,'FontWeight','bold');
114 hold off
115
116
117 % object 3 & 4
118 x3 = centroids(3,1);
119 y3 = centroids(3,2);
120 x4 = centroids(4,1);
121 y4 = centroids(4,2);

```

```
122 figure(9), imshow(I,[])
123 hold on
124 plot([x3;x4], [y3;y4], 'y','LineWidth',3)
125
126 d_px = sum(([x3;y3]-[x4;y4]).^2).^0.5;
127 res = 1.4798;
128 d_mm = d_px/res;
129 text((x3+x4+20)/2,(y3+y4)/2,strcat(['d = ',num2str(d_px),' px']),'Color','y',...
130     'FontSize',14,'FontWeight','bold');
130 text((x3+x4+20)/2,(y3+y4+40)/2,strcat(['d = ',num2str(d_mm),'...
131     'mm']),'Color','y',...
131     'FontSize',14,'FontWeight','bold');
132 hold off
```

Referensi : <https://pemrogramanmatlab.com/2015/10/22/cara-mengukur-jarak-antara-dua-objek-dalam-citra/comment-page-1/>

Nama : Shabila Fitri Aulia

Nim : 202420024

Mata Kuliah : Advanced Database

Dosen : Tri Basuki Kurniawan , S.Kom., M.Eng. Ph.D

**Soal :**

Silahkan cari satu tutorial yang membahas tentang pengukuran jarak antar data. Buat ringkasan dari tutorial tersebut dan tulis dalam format ms word, lalu kumpulkan sebelum batas waktu. Sertakan sumber link tutorial yang diringkas.

Jawaban :

Tutorial pengukuran jarak antar data terbagi menjadi 5,yakni :

1. Role of Distance Measures
2. Hamming Distance
3. Euclidean Distance
4. Manhattan Distance (Taxicab or City Block)
5. Minkowski Distance

Pada pembahasan kali ini, saya akan membahas mengenai hamming distance dengan referensi dari : <https://www.youtube.com/watch?v=vv3ru3R8qCk>

Hamming distance dari dua string adalah jumlah simbol dari kedua string yang berbeda. Sebagai contoh Hamming distance antara string ‘toned’ dan ‘roses’ adalah 3. Hamming Distance juga digunakan untuk mengukur jarak antar dua string binary misalnya jarak antara 10011101 dengan 10001001 adalah 2. Berdasarkan dari sumber yang saya pelajari, terdapat contoh kasus sebagai berikut :

1. Misal terdapat 2 buah string ACTUS dan ACKUC dengan nilai binary 10111 dan 10001. Untuk mencari jarak data maka diperlukan 2 substitusi dari nilai T menjadi K dan S menjadi C. Dengan kata lain jarak hamming distance diantara kedua string tersebut adalah 2.
2. Nilai binary dari kedua string juga memiliki nilai hamming distance 2.

Berdasarkan, materi diatas dapat disimpulkan bahwa untuk menghitung nilai hamming distance, panjang nilai karakter harus sama maka untuk menentukan hamming distance dapat dilihat dari perubahan substitusi karakter. Untuk perhitungan nilai binary, jika nilai binary sama maka hasil = 0, dan jika nilai binary berbeda maka hasil =1 dan penjumlahan nilai hasil sama dengan nilai jarak hamming distance.

Nama : Siti Ratu Delima  
Nim : 202420025  
Matkul : ADVANCED DATABASE (MTIK112)

## Ringkasan tentang pengukuran jarak

### Disini saya akan menggunakan 2 referensi tutorial ringkasan.

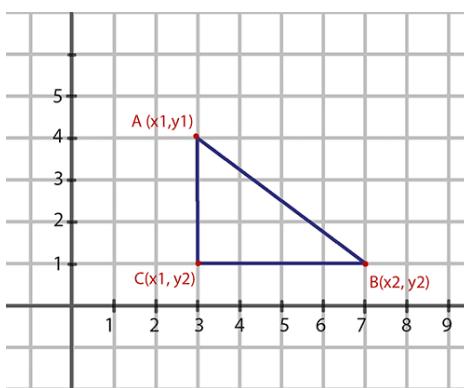
Data clustering salah satu teknik data mining yang bertujuan untuk mengidentifikasi sekelompok obyek yang mempunyai kemiripan karakteristik tertentu yang dapat dipisahkan dengan kelompok obyek lainnya, sehingga obyek yang berada dalam kelompok yang sama relatif lebih homogen daripada obyek yang berada pada kelompok yang berbeda. Tujuan Analisis Cluster Analisis cluster dapat diterapkan pada bidang apa saja. Namun pemakaian teknik ini lebih familiar pada bidang pemasaran karena memang salah satu kegiatan.

Ada 2 konsep engukuran analisis cluster yaitu:

1. konsep pengukuran jarak (distance) pengukuran jarak (distance type measure) digunakan untuk data-data yang bersifat matriks, sedangkan pengukuran kesesuaian (matching type measure) digunakan untuk data-data yang bersifat kualitatif.
2. kesamaan (similarity). adalah ukuran kedekatan. Konsep ini penting karena pengelompokan pada analisis cluster didasarkan pada kedekatan.

Data tersebut dijadikan matriks data mentah berukuran  $m \times p$ . Matrik tersebut ditransformasikan ke dalam bentuk matriks similaritas (kemiripan) berupa  $n \times n$  yang dihitung berdasarkan pasangan-pasangan obyek  $p$  peubah. (<https://www.slideshare.net/triyulianto182/modul-clustering-data-mining-modul-clustering>)

Dan juga, teknik pengukuran jarak euclidean distance adalah perhitungan jarak dari 2 buah titik dalam Euclidean space. Euclidean space diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. (<https://youtu.be/tYyCUnF1yd8>)



Bawa ketika jarak di ilustrasikan bawa ada 2 titik jarak diukur, parameter yang digunakan bisa jadi V dimensi ada beberapa 2 V dimensi: X=2 Y=2 untuk memudahkan.

1. Jarak 2 dimensi Terdapat suatu studi kasus begini, saya memiliki teman bernama delima yang berusia 22 tahun dengan berat badan 67 kg. Teman saya satu lagi, ratu berusia 21 tahun

dengan berat badan 70 kg. Sedangkan saya sendiri berumur 22 tahun dengan berat badan 68kg. dan ipk delima 3.65. ipk ratu 4.00 sedangkan saya 3.68. Pertanyaannya, berdasarkan variabel usia dan berat badan, ipk saya mirip delima atau ratu?

= Jawaban nya adalah pendekatan jarak euclidean. Anggap saja variabel usia adalah x dan variabel berat badan adalah y. Maka, implementasinya seperti ini:

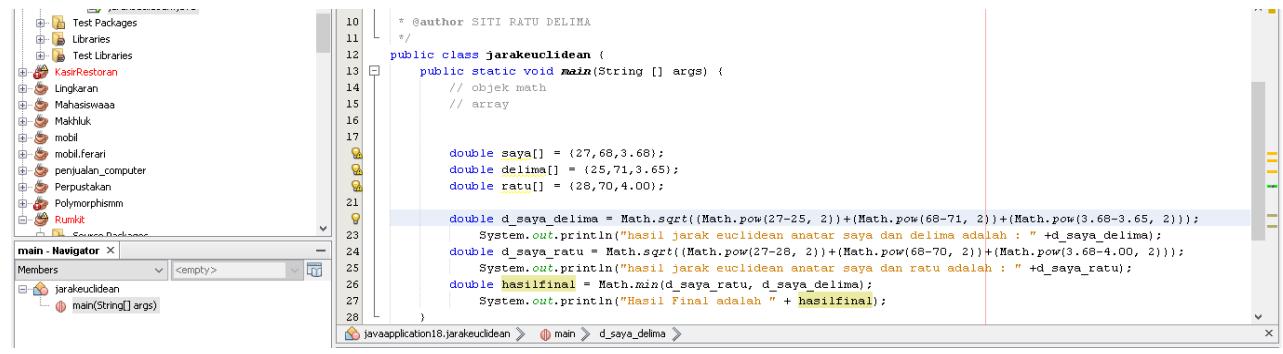
$$d(\text{delima, saya}) = (25-27)^2 + (65-71)^2 = 1+4 = 5. \text{ Hasil dari akar } 5 \text{ adalah } 3,60$$

$$d(\text{ratu, saya}) = (28-27)^2 + (70-68)^2 = 1+9 = 10. \text{ Hasil dari akar } 10 \text{ adalah } 2,25$$

siapa yang paling mirip nilai eculidean berarti dia sama jarak pengukurannya.

Jadi disini saya akan mempraktekan sendiri pengukuran jarak euclidean dengan menggunakan netbeans.

Ini adalah variabel x,y,z

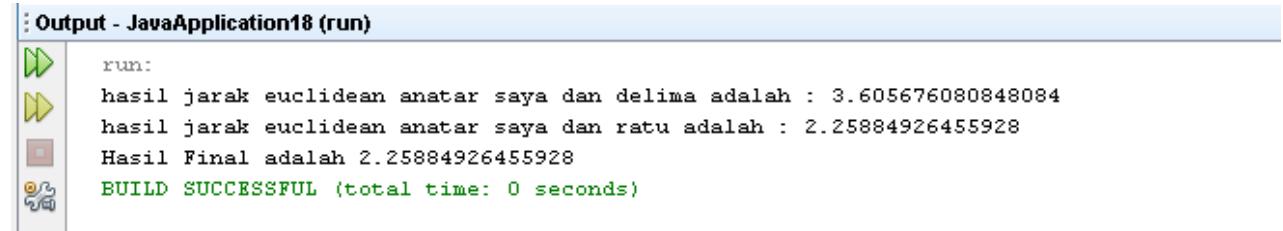


The screenshot shows the NetBeans IDE interface. On the left, there's a file tree with various Java files and a 'main - Navigator' window showing the class 'jarakeuclidean' and its main method. The central area contains the Java code for calculating distances:

```
* @author SITI RATU DELIMA
*/
public class jarakeuclidean {
    public static void main(String [] args) {
        // objek math
        // array

        double saya[] = {27,68,3.68};
        double delima[] = {25,71,3.65};
        double ratu[] = {28,70,4.00};

        double d_saya_delima = Math.sqrt((Math.pow(27-25, 2)+(Math.pow(68-71, 2))+(Math.pow(3.68-3.65, 2)));
        System.out.println("hasil jarak euclidean antar saya dan delima adalah : "+d_saya_delima);
        double d_saya_ratu = Math.sqrt((Math.pow(27-28, 2)+(Math.pow(68-70, 2))+(Math.pow(3.68-4.00, 2)));
        System.out.println("hasil jarak euclidean antar saya dan ratu adalah : "+d_saya_ratu);
        double hasilfinal = Math.sqrt(d_saya_ratu, d_saya_delima);
        System.out.println("Hasil Final adalah " + hasilfinal);
    }
}
```



The screenshot shows the 'Output - JavaApplication18 (run)' window. It displays the command 'run:' followed by the execution results:

```
hasil jarak euclidean antar saya dan delima adalah : 3.605676080848084
hasil jarak euclidean antar saya dan ratu adalah : 2.25884926455928
Hasil Final adalah 2.25884926455928
BUILD SUCCESSFUL (total time: 0 seconds)
```

Kalau hasil jarak euclidean ternyata bisa dibandingkan, siapa yang paling mirip siapa yang paling kecil. Kalau angka yang seperti ini berarti mirip, dan otomatiskan java akan dibandingkan jarak dan pengukuran yang dimana hasil final yang di perkecilkan.

## TUTORIAL PENGUKURAN JARAK ANTAR DATA

NAMA : SURTA WIJAYA

NIM : 202420014

<https://www.advernesia.com/blog/data-science/pengertian-dan-cara-kerja-algoritma-k-nearest-neighbours-KNN/>

Menghitung jarak antar data memegang peranan yang sangat penting di dalam *data mining*. Ada beberapa metode yang digunakan dalam pengukuran jarak antar data. Salah satu caranya yaitu dengan menggunakan metode K-nearest neighbors atau KNN. KNN adalah algoritma yang berfungsi untuk melakukan klasifikasi suatu data berdasarkan data pembelajaran (*train data sets*), yang diambil dari k tetangga terdekatnya (*nearest neighbors*). Dengan k merupakan banyaknya tetangga terdekat. K-nearest neighbors melakukan klasifikasi dengan proyeksi data pembelajaran pada ruang berdimensi banyak. Ruang ini dibagi menjadi bagian-bagian yang merepresentasikan kriteria data pembelajaran. Setiap data pembelajaran direpresentasikan menjadi titik-titik **c** pada ruang dimensi banyak.

- **Klasifikasi Terdekat (Nearest Neighbor Classification)**

**Data baru** yang diklasifikasi selanjutnya diproyeksikan pada ruang dimensi banyak yang telah memuat titik-titik **c** data pembelajaran. Proses klasifikasi dilakukan dengan mencari titik **c** terdekat dari **c-baru** (*nearest neighbor*). Teknik pencarian tetangga terdekat yang umum dilakukan dengan menggunakan formula jarak euclidean. Berikut beberapa formula yang digunakan dalam algoritma knn.

<https://blogs.itb.ac.id/anugraha/2014/09/10/theory-pengukuran-jarak/>

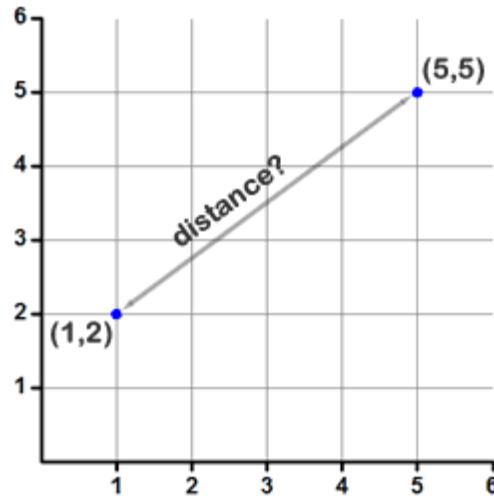
- **Euclidean Distance**

Euclidean distance adalah perhitungan jarak garis lurus antar 2 buah titik dalam Euclidean space. Untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini berkaitan dengan Teorema Phytagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tapi juga sederhana jika diterapkan pada dimensi yang lebih tinggi.

- a. **Pada 1 dimens**

Semisal ingin menghitung jarak Euclidean 1 dimensi. Titik pertama adalah 4, titik kedua adalah -10. Caranya adalah kurangkan -10 dengan 4. sehingga menghasilkan -14. Cari nilai absolut dari nilai -14 dengan cara mempangatkannya sehingga mendapat nilai 196. Kemudian diakarkan sehingga mendapatkan nilai 14. Sehingga jarak euclidean dari 2 titik tersebut adalah 14.

- b. **Pada 2 dimensi**



Koordinat jarak

Caranya hampir sama. Misalkan titik pertama mempunyai kordinat (1,2). Titik kedua ada di kordinat (5,5). Caranya adalah kurangkan setiap kordinat titik kedua dengan titik yang pertama. Yaitu, (5-1,5-2) sehingga menjadi (4,3). Kemudian pangkatkan masing-masing sehingga memperoleh (16,9). Kemudian tambahkan semuanya sehingga memperoleh nilai  $16+9 = 25$ . Hasil ini kemudian diakarkan menjadi 5. Sehingga jarak eucliennya adalah 5.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Rumus Euclid

Sehingga dari Formula diatas kita dapat implementasi menjadi :

$$\boxed{Jarak = \sqrt{(Lat_1 - Lat_2)^2 + (Long_1 - Long_2)^2}}$$

Rumus Jarak Euclidean

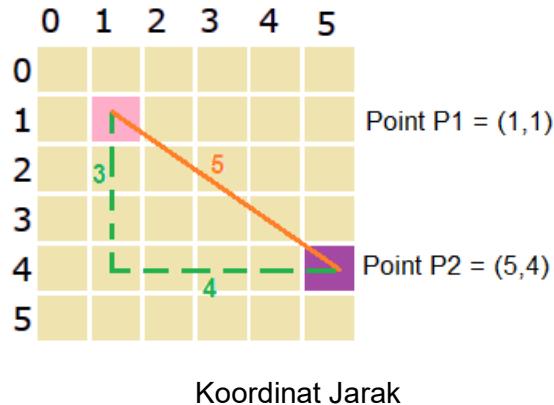
#### ➤ **Manhattan distance**

Manhattan distance digunakan untuk menghitung perbedaan absolut (mutlak) antara koordinat sepasang objek. Rumus yang digunakan adalah sebagai berikut:

$$Mdist = |x_2 - x_1| + |y_2 - y_1|$$

Rumus Manhattan Distance

Contoh perhitungannya adalah sebagai berikut :



Koordinat Jarak

Diketahui bahwa titik P1 memiliki koordinat (1,1), dan titik P2 memiliki koordinat (5,4), untuk menghitung jarak antara titik P2 dan P1 , yaitu dengan cara =  $(5-1) + (4-1) = 4+3 = 7$  .

<http://mafisamin.blog.ugm.ac.id/files/2014/06/c10-deteksi-error-dan-koreksi-lanjutan.pdf>

#### ➤ **Hamming distance**

Dalam block coding transmisi data dilakukan dengan membagi deretan bit menjadi block yang terdiri dari sejumlah bit ( $k$ ) yang disebut dengan words serta menambahkan sejumlah bit tambahan (redundancy/ $r$ ) ke dalam setiap block sehingga panjang dari block yang akan ditransmisikan menjadi:  $n = k + r$  disebut dengan codewords.. Salah satu konsep penting dalam blok kode untuk transmisi data adalah jarak hamming (hamming distance), yang diartikan sebagai jumlah perbedaan antara dua buah deretan bit (words) yang mempunyai ukuran sama. Jika terdapat 2 buah deretan bit misalnya  $x$  dan  $y$  maka jarak hamming dapat disimbolkan dengan:

$d(x,y)$  Jarak hamming cukup mudah untuk diimplementasikan, yaitu dengan menggunakan operasi XOR pada dua buah words, dan menghitung jumlah bit 1 dari hasil operasi.

Contoh : Tentukan jarak hamming dari pasangan words berikut.

- 1)  $d(000, 011)$  akan menghasilkan 2, karena terdapat hasil 011 (dimana jumlah bit 1 berjumlah 2)
- 2)  $d(10101, 11110)$  menghasilkan 3, karena terdapat hasil 01011 (dimana jumlah bit 1 berjumlah 3)

Jarak hamming akan menghasilkan bilangan lebih besar dari 0 (nol). Jarak hamming juga dapat digunakan untuk melakukan proses deteksi dan koreksi error pada data yang ditransmisikan. Jarak minimum hamming distance merupakan jarak hamming terkecil antara semua kemungkinan pasangan words.

Tabel 1. Pasangan Word

<i>Datawords</i>	<i>Codewords</i>
00	000
01	011
10	101
11	110

Pada block code pada Table 1, dapat ditentukan minimum hamming distance sebagai berikut:

$$\begin{aligned}
 d(000, 011) &= 2 \\
 d(000, 101) &= 2 \\
 d(000, 110) &= 2 \\
 d(011, 101) &= 2 \\
 d(011, 110) &= 2 \\
 d(101, 110) &= 2
 \end{aligned}$$

sehingga dapat ditentukan bahwa  $d_{MIN} = 2$ . Selanjutnya dapat dilakukan proses yang sama untuk menentukan  $d_{MIN}$  dari Tabel 2.

Tabel 2 Pasangan Codeword 5 digit

<i>Dataword</i>	<i>Codeword</i>
00	00000
01	01011
10	10101
11	11110

Hasil dari perhitungan hamming distance diperoleh hasil seperti Gambar 1, sehingga dapat disimpulkan  $d_{MIN} = 3$  karena nilai terkecil dari hamming distance adalah 3.

$d(00000, 01011) = 3$	$d(00000, 10101) = 3$	$d(00000, 11110) = 4$
$d(01011, 10101) = 4$	$d(01011, 11110) = 3$	$d(10101, 11110) = 3$

Gambar 1 Hasil Perhitungan Hamming Distance

Untuk menjamin deteksi hingga  $s$  kesalahan dalam semua kasus, minimum jarak hamming dalam sebuah block code harus  $d_{MIN} = s + 1$ . Artinya minimum hamming distance hanya mampu mendeteksi error  $d_{MIN} - 1$  pada data yang ditransmisikan.

Jika terdapat  $d_{MIN} = 2$ , hal ini mengindikasikan bahwa block code menjamin hanya terjadi single-bit error. Sebagai contoh pada Tabel 1, data ke-3 pada codeword 101 ditransmisikan dan satu error terjadi, maka codeword yang diterima tidak sama dengan codeword yang valid di dalam

daftar codeword pada Tabel 1. Jika terjadi 2 error, dapat saja codeword yang diterima cocok dengan codeword valid tapi dampaknya adalah error tidak dapat dideteksi. Contoh lain pada Tabel 2, nilai dari  $d_{MIN} = 3$ , sehingga block code hanya mampu mendeteksi error sebanyak 2 bit. Sebagai contoh:

Data 00000 ditransmisikan kemudian terdapat 2 bit error misal 00011 atau 01100, atau 11000, atau 10100

Maka dapat dipastikan data yang diterima merupakan data yang tidak valid sehingga receiver mendeteksi terjadinya error pada data yang dikirim. Tetapi jika data yang diterima terjadi error lebih dari 2 akan mengakibatkan error tidak terdeteksi. Kondisi lain jika terjadi error sebanyak 3 buah, maka dapat mengakibatkan terjadinya error yang tidak terdeteksi, karena data yang error bisa jadi merupakan codeword yang lain.

#### ➤ Minkowski Distance

Minkowski Distance merupakan generalisasi dari Euclidean distance. Minkowski distance yaitu sebuah metrik dalam ruang vektor di mana suatu norma didefinisikan (normed vector space) sekaligus dianggap sebagai generalisasi dari Euclidean distance dan Manhattan distance. Dalam pengukuran jarak objek menggunakan minkowski distance biasanya digunakan nilai  $p$  adalah 1 atau 2. Berikut rumus yang digunakan menghitung jarak dalam metode ini.

$$d(x, y) = \left( \sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}$$

Rumus minkowski distance

Jarak Minkowski adalah metrik jarak umum. Di sini digeneralisasikan berarti kita dapat memanipulasi rumus di atas untuk menghitung jarak antara dua titik data dengan cara yang berbeda. Seperti disebutkan di atas, kita dapat memanipulasi nilai  $p$  dan menghitung jarak dengan tiga cara berbeda-

$p = 1$ , Jarak Manhattan

$p = 2$ , Jarak Euclidean

$p = \infty$ , Jarak Chebychev

#### • Banyaknya k Tetangga Terdekat

Untuk menggunakan algoritma k nearest neighbors, perlu ditentukan banyaknya k tetangga terdekat yang digunakan untuk melakukan klasifikasi data baru. Banyaknya k, sebaiknya merupakan angka ganjil, misalnya  $k = 1, 2, 3$ , dan seterusnya. Penentuan nilai k dipertimbangkan berdasarkan banyaknya data yang ada dan ukuran dimensi yang dibentuk oleh data. Semakin banyak data yang ada, angka k yang dipilih sebaiknya semakin rendah. Namun, semakin besar ukuran dimensi data, angka k yang dipilih sebaiknya semakin tinggi.

- **Algoritma K-Nearest Neighbors**

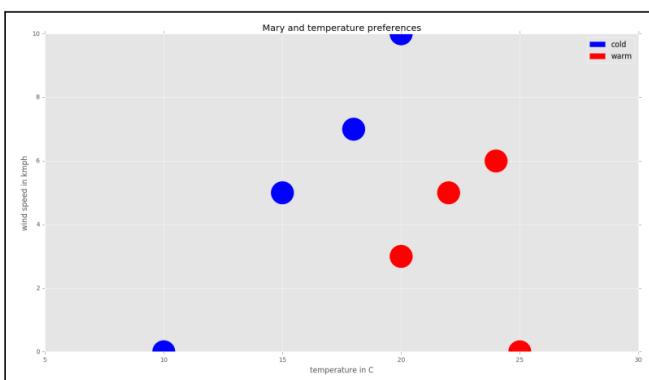
1. Tentukan k bilangan bulat positif berdasarkan ketersediaan data pembelajaran.
2. Pilih tetangga terdekat dari data baru sebanyak k.
3. Tentukan klasifikasi paling umum pada langkah (ii), dengan menggunakan frekuensi terbanyak.
4. Keluaran klasifikasi dari data sampel baru.

- **Contoh Aplikasi K Nearest Neighbors**

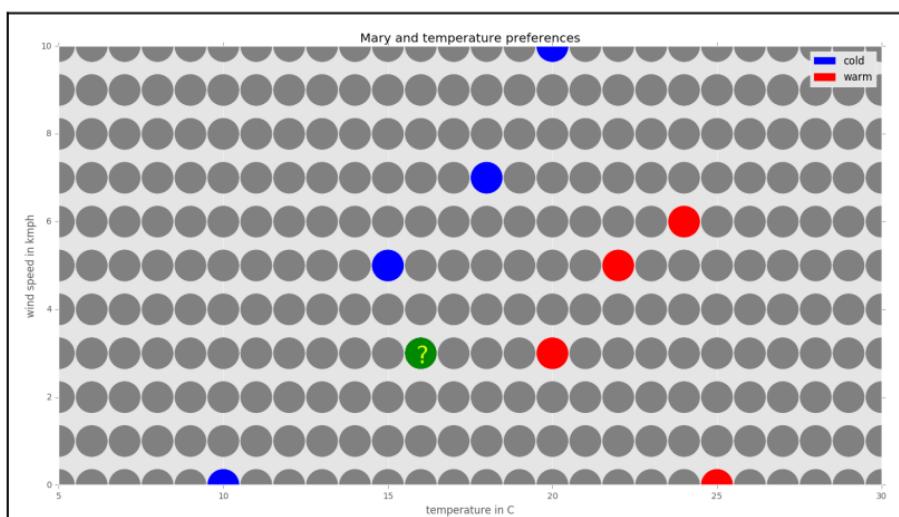
Pada contoh ini, dilakukan klasifikasi suhu udara berdasarkan persepsi seseorang yang bernama Marry. Adapun klasifikasi suhu udara terdiri dari 2 persepsi yaitu **Panas** dan **Dingin**. Persepsi ini dapat diukur berdasarkan 2 variabel yaitu **temperatur dalam derajat celcius** dan **kecepatan angin dalam km/h**. Diperoleh data berikut,

Temperatur Udara (°C)	Kecepatan Angin (km/jam)	Klasifikasi atau Persepsi Marry
10	0	Dingin
25	0	Panas
15	5	Dingin
20	3	Panas
18	7	Dingin
20	10	Dingin
22	5	Panas
24	6	Panas

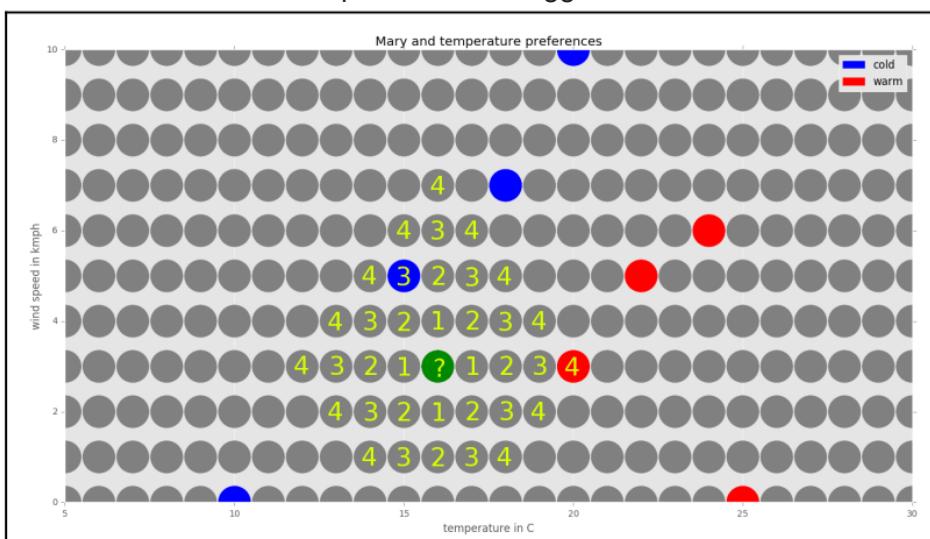
Untuk contoh ini terbentuk ruang dimensi 2, yang berisi 2 kriteria yaitu **temperatur udara** dan **kecepatan angin**.



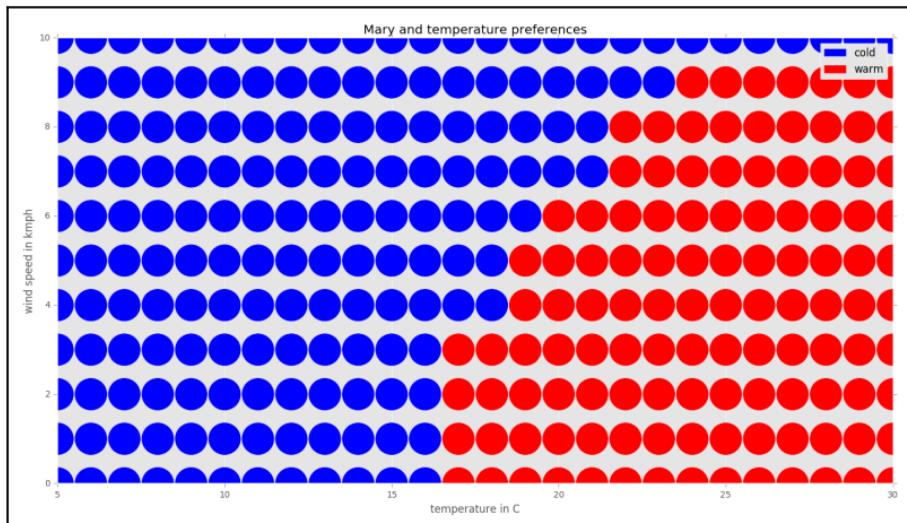
Pada proyeksi di atas sumbu vertikal adalah kecepatan angin, sumbu horizontal adalah temperatur suhu, warna biru adalah dingin, dan warna merah adalah panas. Dari proyeksi di atas, dapat dilakukan klasifikasi data baru. Misalnya, Bagaimana persepsi Mary saat temperatur udara 16°C dan kecepatan angin 3 km/jam.



Proses pencarian tetangga terdekat



Dapat diketahui tetangga terdekatnya adalah titik  $c$  **dingin** dengan temperatur  $15^{\circ}\text{C}$  dan kecepatan angin  $5 \text{ km/jam}$ . Jadi berdasarkan pemilihan  $k = 1$ , klasifikasinya adalah **dingin**. Dengan melakukan proses di atas terhadap semua titik, diperoleh proyeksi klasifikasi berikut.



Catatan: Untuk pemilihan  $k$  lainnya, hasil klasifikasi ditentukan dengan frekuensi terbanyak. Misalnya  $k = 3$ , dengan titik terdekat dingin, panas, dingin. Hasil klasifikasi data baru tersebut adalah dingin.

Nama : Trada Ayang Pratiwi  
NIM : 202420020  
Mata Kuliah : Advance Database  
Dosen Pengasuh : Tri Basuki Kurniawan, S.Kom., M.eng., Ph.D.  
Judul Tugas : Ringkasan Video Tutorial Pengukuran Jarak Antar Data  
Link Video : <https://www.youtube.com/watch?v=mBIxKf2uTD4&t=141s>

Metode pengukuran jarak digunakan untuk memudahkan kita mengukur jarak yang sebenarnya seperti panjang, keliling dan luas suatu wilayah.

Polyline digunakan untuk mengukur jarak panjang yang sebenarnya suatu wilayah dengan cara menentukan dua titik, misalkan titik A dan titik B.

Contoh : mengukur panjang yang sebenarnya landasan bandar udara yang ada di Kota Padang menggunakan ArchGIS dengan mengambil salah satu image bandar udara menggunakan salah satu citra satelit.

Polygon digunakan untuk mengukur keliling dan luas yang sebenarnya suatu wilayah.

Contoh : mengukur keliling dan luas yang sebenarnya bandar udara di kota Padang menggunakan ArchGIS dengan mengambil salah satu image bandar udara menggunakan salah satu citra satelit

Catatan :

1. Sebelum mengukur panjang tentukanlah dua titik yang akan diukur, misalkan titik A dan titik B.
2. Sebelum mengukur keliling dan luas tentukanlah luas yang akan diukur dengan cara membuat patokan pada batas-batas wilayah yang akan kita ukur
3. Bebas menggunakan citra satelit manapun.

NAMA : VERO FALORIS

NIM : 202420032

KELAS : MTI 23 REGULER A

MK : *ADVANCED DATABASE*

DOSEN : TRI BASUKI KURNIAWAN, S.KOM., M.ENG., PH.D

**SOAL :**

Carilah satu tutorial yang membahas tentang, pengukur jarak antar data. Buatlah ringkasan dari Tutorial Tersebut dan tulis dalam Format Ms. Word, lalu kumpulkan sebelum batas waktu, sertakan sumber link tutorial yang diringkas.

**JAWAB :**

Rumus Jarak :  $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

1. Ambillah koordinat dari dua titik yang ingin Anda cari jaraknya. Sebutlah salah satu titik sebagai Titik 1 ( $x_1, y_1$ ) dan titik lainnya sebagai Titik 2 ( $x_2, y_2$ ). Tidak masalah titik mana yang menjadi titik 1 atau 2 selama Anda tetap konsisten dalam memberi label (1 dan 2) saat menyelesaikan soal.[1]  
 $x_1$  adalah koordinat horizontal (searah dengan sumbu x) dari Titik 1, dan  $x_2$  adalah koordinat horizontal dari Titik 2.  $y_1$  adalah koordinat vertikal (searah dengan sumbu y) dari Titik 1, dan  $y_2$  adalah koordinat vertikal dari Titik 2.  
Misalnya, gunakan titik-titik (3,2) dan (7,8). Jika (3,2) adalah  $(x_1, y_1)$ , maka (7,8) adalah  $(x_2, y_2)$ .

2. Ketahui rumus jarak. Rumus ini menghitung panjang garis yang terbentang di antara dua titik: Titik 1 dan Titik 2. Jarak lininya merupakan akar kuadrat dari kuadrat jarak horizontal ditambah kuadrat jarak vertikal di antara kedua titik. Singkatnya, jarak linier merupakan akar kuadrat dari:  $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
3. Carilah jarak horizontal dan vertikal di antara dua titik. Pertama, kurangkan  $y_2 - y_1$  untuk mencari jarak vertikalnya. Kemudian, kurangkan  $x_2 - x_1$  untuk mencari jarak horizontalnya. Jangan khawatir jika pengurangan menghasilkan angka negatif. Langkah selanjutnya adalah menguadratkan nilai-nilai ini, dan penguadratan selalu menghasilkan angka bulat positif.[3]
  - a. Carilah jarak yang searah dengan sumbu y. Untuk contoh titik-titik (3,2) dan (7,8), dengan (3,2) sebagai Titik 1 dan (7,8) sebagai Titik 2:  $(y_2 - y_1) = 8 - 2 = 6$ . Ini berarti ada enam satuan jarak di antara kedua titik ini pada sumbu y.
  - b. Carilah jarak yang searah dengan sumbu x. Untuk contoh titik-titik (3,2) dan (7,8):  $(x_2 - x_1) = 7 - 3 = 4$ . Ini berarti ada empat satuan jarak yang memisahkan kedua titik itu pada sumbu x.

4. Kuadratkan kedua nilainya. Ini berarti Anda akan menguadratkan jarak pada sumbu x ( $x_2 - x_1$ ), dan Anda akan menguadratkan jarak pada sumbu y ( $y_2 - y_1$ ) secara terpisah.

$$\sqrt{6^2} = 36$$

$$\sqrt{4^2} = 16$$

5. Jumlahkan nilai kuadratnya. Penjumlahan ini akan menghasilkan kuadrat jarak linier diagonal di antara kedua titik Anda. Dalam contoh titik-titik (3,2) dan (7,8), kuadrat dari  $(7 - 3)$  adalah 16, dan kuadrat dari  $(8 - 2)$  adalah 36.  $36 + 16 = 52$ .
- 6.
7. Carilah akar kuadrat dari persamaan. Ini adalah langkah terakhir dalam persamaan. Jarak linier di antara kedua titik merupakan akar kuadrat dari jumlah nilai kuadrat jarak pada sumbu x dan jarak pada sumbu y.[4]

Untuk melanjutkan contoh di atas: jarak antara (3,2) dan (7,8) adalah akar (52), atau sekitar 7,21 satuan.

Tips :

Tidak masalah jika Anda mendapatkan angka negatif setelah mengurangkan  $y_2 - y_1$  atau  $x_2 - x_1$  karena Anda akan selalu mendapatkan jarak yang bernilai positif sebagai jawaban saat Anda memangkatkan selisih keduanya.

<https://www.youtube.com/watch?v=0IOEPcAHgi4>

<https://id.wikihow.com/Mencari-Jarak-antara-Dua-Titik>

# Mengukur Jarak Data

Salah satu tantangan dalam era ini dengan database yang memiliki banyak tipe data. Mengukur jarak adalah komponen utama dalam algoritma clustering berbasis jarak. Algoritma seperti Algoritma Partisioning misal K-Mean, K-medoid dan fuzzy c-mean dan rough clustering bergantung pada jarak untuk melakukan pengelompokan

Sebelum menjelaskan tentang beberapa macam ukuran jarak, kita mendefinisikan terlebih dahulu yaitu  $v_1, v_2$  menyatakan dua vektor yang menyatakan  $v_1 = x_1, x_2, \dots, x_n, v_2 = y_1, y_2, \dots, y_n, v_1 = x_1, x_2, \dots, x_n, v_2 = y_1, y_2, \dots, y_n$ , dimana  $x_i, y_i$  disebut atribut. Ada beberapa ukuran similaritas atau ukuran jarak, diantaranya **Minkowski Distance**

Kelompok Minkowski diantaranya adalah Euclidean distance dan Manhattan distance, yang menjadi kasus khusus dari Minkowski distance. Minkowski distance dinyatakan dengan

$$d_{min} = (\sum_{i=1}^n |x_i - y_i|^m)^{1/m}, m \geq 1$$

dimana  $m$  adalah bilangan riil positif dan  $x_i$  dan  $y_i$  adalah dua vektor dalam ruang dimensi  $n$ . Implementasi ukuran jarak Minkowski pada model clustering data atribut dilakukan normalisasi untuk menghindari dominasi dari atribut yang memiliki skala data besar.

## **Manhattan distance**

Manhattan distance adalah kasus khusus dari jarak Minkowski distance pada  $m = 1$ . Seperti Minkowski Distance, Manhattan distance sensitif terhadap outlier. Jika ukuran ini digunakan dalam algoritma clustering, bentuk cluster adalah hyper-rectangular. Ukuran ini didefinisikan dengan

$$d_{man} = \sqrt{\sum_{i=1}^n |x_i - y_i|}$$

## **Euclidean distance**

Jarak yang paling terkenal yang digunakan untuk data numerik adalah jarak Euclidean. Ini adalah kasus khusus dari jarak Minkowski ketika  $m = 2$ . Jarak Euclidean berkinerja baik ketika digunakan untuk kumpulan data cluster kompak atau terisolasi. Meskipun jarak Euclidean sangat umum dalam pengelompokan, ia memiliki kelemahan: jika dua vektor data tidak memiliki nilai atribut yang sama, kemungkinan memiliki jarak yang lebih kecil daripada pasangan vektor data lainnya yang mengandung nilai atribut yang sama. Masalah lain dengan jarak Euclidean sebagai fitur skala terbesar akan mendominasi yang lain. Normalisasi fitur kontinu adalah solusi untuk mengatasi kelemahan ini.

### **Average Distance**

Berkenaan dengan kekurangan dari Jarak Euclidian Distance diatas, rata rata jarak adalah versi modifikasi dari jarak Euclidian untuk memperbaiki hasil. Untuk dua titik  $x, y \in \mathbb{R}^n$  dalam ruang dimensi  $n$ , rata-rata jarak didefinisikan dengan

$$d_{ave} = \frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2$$

### **Weighted euclidean distance**

Jika berdasarkan tingkatan penting dari masing masing atribut ditentukan, maka Weighted Euclidean distance adalah modifikasi lain dari jarak Euclidean distance yang dapat digunakan. Ukuran ini dirumuskan dengan

$$d_{we} = \left( \sum_{i=1}^n w_i (x_i - y_i)^2 \right)^{1/2}$$
 dimana  $w_i$  adalah bobot yang diberikan pada atribut ke  $i$ .

### **Chord distance**

Chord distance adalah salah satu ukuran jarak modifikasi Euclidean distance untuk mengatasi kekurangan dari Euclidean distance. Ini dapat dipecahkan juga dengan menggunakan skala pengukuran yang baik. Jarak ini dapat juga dihitung dari data yang tidak dinormalisasi. Chord distance didefinisikan dengan

$$d_{chord} = \sqrt{2 - 2 \sum_{i=1}^n x_i y_i / \|x\|_2 \|y\|_2}$$

dimana  $\|x\|_2$  adalah L<sub>2</sub>-norm  $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$

### **Mahalanobis distance**

Mahalanobis distance berdasarkan data berbeda dengan Euclidean dan Manhattan distances yang bebas antara data dengan data yang lain. Jarak Mahalanobis yang teratur dapat digunakan untuk mengekstraksi hyperellipsoidal clusters. Jarak Mahalanobis dapat mengurangi distorsi yang disebabkan oleh korelasi linier antara fitur dengan menerapkan transformasi pemutihan ke data atau dengan menggunakan kuadrat Jarak mahalanobis. Mahalanobis distance dinyatakan dengan

$$d_{mah} = \sqrt{(x - y)^T S^{-1} (x - y)}$$

dimana  $S$  adalah matrik covariance data.

### **Cosine measure**

Ukuran Cosine similarity lebih banyak digunakan dalam similaritas dokumen dan dinyatakan dengan

$$\text{Cosine}(x, y) = \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}}$$

dimana  $\|y\|_2$  adalah Euclidean norm dari vektor  $y=(y_1, y_2, \dots, y_n)$ .  $y=(y_1, y_2, \dots, y_n)$  didefinisikan dengan  $\|y\|_2 = \sqrt{y_{11}^2 + y_{22}^2 + \dots + y_{nn}^2}$

### **Pearson correlation**

Pearson correlation banyak digunakan dalam data expresi gen. Ukuran similaritas ini menghitung similaritas antara dua bentuk pola expresi gen. Pearson correlation didefinisikan dengan

$$\text{Pearson}(x, y) = \frac{\sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^n (x_i - \mu_x)^2} \sqrt{\sum_{i=1}^n (y_i - \mu_y)^2}}$$

The Pearson correlation kelemahannya adalah sensitif terhadap outlier

### **Mengukur Jarak Atribut Binary**

Mari kita lihat similaritas dan desimilarity untuk objek yang dijelaskan oleh atribut biner simetris atau asimetris. Atribut biner hanya memiliki dua status: 0 dan 1. Contoh atribut perokok menggambarkan seorang pasien, misalnya, 1 menunjukkan bahwa pasien merokok, sedangkan 0 menunjukkan pasien tidak merokok. Memperlakukan atribut biner sebagai atribut numerik tidak diperkenankan. Oleh karena itu, metode khusus untuk data biner diperlukan untuk membedakan komputasi.

Jadi, bagaimana kita bisa menghitung ketidaksamaan antara dua atribut biner? "Satu pendekatan melibatkan penghitungan matriks ketidaksamaan dari data biner yang diberikan. Jika semua atribut biner dianggap memiliki bobot yang sama, kita memiliki tabel kontingensi  $2 \times 2 \times 2$  di mana  $qq$  adalah jumlah atribut yang sama dengan 1 untuk kedua objek  $ii$  dan  $jj$ ,  $rr$  adalah jumlah atribut yang sama dengan 1 untuk objek  $ii$  tetapi 0 untuk objek  $jj$ ,  $ss$  adalah jumlah atribut yang sama dengan 0 untuk objek  $ii$  tetapi 1 untuk objek  $jj$ , dan  $tt$  adalah jumlah atribut yang sama dengan 0 untuk kedua objek  $ii$  dan  $jj$ . Jumlah total atribut adalah  $pp$ , di mana  $p=q+r+s+tp=q+r+s+t$

Inginlah bahwa untuk atribut biner simetris, masing-masing nilai bobot yang sama. Dissimilarity yang didasarkan pada atribut asymmetric binary disebut symmetric binary dissimilarity. Jika objek  $i$  dan  $j$  dinyatakan sebagai atribut biner simetris, maka dissimilarity antar  $ii$  dan  $jj$  adalah

$$d(i,j) = r + sq + r + s + td(i,j) = r + sq + r + s + t$$

Untuk atribut biner asimetris, kedua kondisi tersebut tidak sama pentingnya, seperti hasil positif (1) dan negatif (0) dari tes penyakit. Diberikan dua atribut biner asimetris, pencocokan keduanya 1 (kecocokan positif) kemudian dianggap lebih signifikan daripada kecocokan negatif. Ketidaksamaan berdasarkan atribut-atribut ini disebut asimetris biner dissimilarity, di mana jumlah kecocokan negatif,  $t$ , dianggap tidak penting dan dengan demikian diabaikan. Berikut perhitungannya

$$d(i,j) = r + sq + r + sd(i,j) = r + sq + r + s$$

Kita dapat mengukur perbedaan antara dua atribut biner berdasarkan pada disimilarity. Misalnya, biner asimetris kesamaan antara objek  $i$  dan  $j$  dapat dihitung dengan

$$\text{sim}(i,j) = \frac{qq+r+s}{qq+r+s} = 1 - d(i,j)$$

Persamaan similarity ini disebut dengan **Jaccard coefficient**

### **Mengukur Jarak Tipe categorical**

*Li, C., & Li, H. (2010). A Survey of Distance Metrics for Nominal Attributes. JSW, 5(11), 1262-1269.*

#### **Overlay Metric**

Ketika semua atribut adalah bertipe nominal, ukuran jarak yang paling sederhana adalah dengan Overlay Metric (OM) yang dinyatakan dengan

$$d(x,y) = \sum_{i=1}^n \delta(a_i(x), a_i(y))$$

dimana  $n$  adalah banyaknya atribut,  $a_i(x)$  dan  $a_i(y)$  adalah nilai atribut ke  $i$  yaitu  $A_i$  dari masing-masing objek  $x$  dan  $y$ ,  $\delta(a_i(x), a_i(y))$  adalah 0 jika  $a_i(x) = a_i(y)$  dan 1 jika sebaliknya.

OM banyak digunakan oleh instance-based learning dan locally weighted learning. Jelas sekali, ini sedikit beruk untuk mengukur jarak antara masing-masing pasangan sample, karena gagal memanfaatkan tambahan informasi yang diberikan oleh nilai atribut nominal yang bisa membantu dalam generalisasi.

#### **Value Difference Metric (VDM)**

VDM dikenalkan oleh Standfill and Waltz, versi sederhana dari VDM tanpa skema pembobotan didefinisikan dengan

$$d(x,y) = \sum_{i=1}^n \sum_{c=1}^C |P(c|a_i(x)) - P(c|a_i(y))|$$

dimana  $C$  adalah banyaknya kelas,  $P(c|a_i(x))P(c|a_i(y))$  adalah probabilitas bersyarat dimana kelas  $x$  adalah  $c$  dari atribut  $A_i$ , yang memiliki nilai  $a_i(x)$ ,  $P(c|a_i(y))P(c|a_i(y))$  adalah probabilitas bersyarat dimana kelas  $y$  adalah  $c$  dengan atribut  $A_i$  memiliki nilai  $a_i(y)$

VDM mengasumsikan bahwa dua nilai dari atribut adalah lebih dekat jika memiliki klasifikasi sama. Pendekatan lain berbasis probabilitas adalah SFM (Short and Fukunaga Metric) yang kemudian dikembangkan oleh Myles dan Hand dan didefinisikan dengan

$$d(x,y) = \sum_{c=1}^C |P(c|x) - P(c|y)|$$

dimana probabilitas keanggotaan kelas diestimasi dengan  $P(c|x)P(c|x)$  dan  $P(c|y)P(c|y)$  didekati dengan Naive Bayes,  
**Minimum Risk Metric (MRM)**

Ukuran ini dipresentasikan oleh Blanzieri and Ricci, berbeda dari SFM yaitu meminimumkan selisih antara kesalahan berhingga dan kesalahan asymptotic. MRM meminimumkan risk of misclassification yang didefinisikan dengan

$$d(x,y)=\sum_{c=1}^C P(c|x)(1-P(c|y))$$

#### Mengukur Jarak Tipe Ordinal

**Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.**

Nilai-nilai atribut ordinal memiliki urutan atau peringkat, namun besarnya antara nilai-nilai berturut-turut tidak diketahui. Contohnya tingkatan kecil, sedang, besar untuk atribut ukuran. Atribut ordinal juga dapat diperoleh dari diskritisasi atribut numerik dengan membuat rentang nilai ke dalam sejumlah kategori tertentu. Kategori-kategori ini disusun dalam peringkat. Yaitu, rentang atribut numerik dapat dipetakan ke atribut ordinal  $f_f$  yang memiliki  $M_f M_f$  state. Misalkan, kisaran suhu atribut skala-skala (dalam Celcius) dapat diatur ke dalam status berikut: -30 hingga -10, -10 hingga 10, 10 hingga 30, masing-masing mewakili kategori suhu dingin, suhu sedang, dan suhu hangat.  $M_M$  adalah jumlah keadaan yang dapat dilakukan oleh atribut ordinal memiliki. State ini menentukan peringkat  $1, \dots, M_f, \dots, M_f$

Perlakuan untuk atribut ordinal adalah cukup sama dengan atribut numerik ketika menghitung dissimilarity antara objek. Misalkan  $f_f$  adalah atribut-atribut dari atribut ordinal dari  $n$  objek. Menghitung dissimilarity terhadap  $f_f$  fitur sebagai berikut:

- Nilai  $f_f$  untuk objek ke- $i$  adalah  $x_{if} x_{if}$ , dan  $f_f$  memiliki  $M_f M_f$  status urutan, mewakili peringkat  $1, \dots, M_f, \dots, M_f$ . Ganti setiap  $x_{if} x_{if}$  dengan peringkatnya,  $r_{if} \in \{1, \dots, M_f\}, r_{if} \in \{1, \dots, M_f\}$
- Karena setiap atribut ordinal dapat memiliki jumlah state yang berbeda, diperlukan untuk memetakan rentang setiap atribut ke  $[0,0, 1.0]$  sehingga setiap atribut memiliki bobot yang sama. Perlakukan normalisasi data dengan mengganti peringkat  $r_{if} r_{if}$  dengan  $z_{if} = r_{if} - 1, M_f - 1$
- Dissimilarity kemudian dihitung dengan menggunakan ukuran jarak seperti atribut numerik dengan data yang baru setelah ditransformasi  $z_{if}$

#### Menghitung Jarak Tipe Campuran

**Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. Journal of artificial intelligence research, 6, 1-34.**

Menghitung ketidaksamaan antara objek dengan atribut campuran yang berupa nominal, biner simetris, biner asimetris, numerik, atau ordinal yang ada pada kebanyakan databases dapat dinyatakan dengan memproses semua tipe atribut secara bersamaan. Salah satu teknik tersebut menggabungkan atribut yang berbeda ke dalam matriks ketidaksamaan

tunggal dan menyatakannya dengan skala interval antar  $[0,0,1.0][0,0,1.0]$ . Misalkan data berisi atribut **pp** tipe campuran. Ketidaksamaan (disimilarity) antara objek **ii** dan **jj** dinyatakan dengan

$$d(i,j)=\sum_{pf=1}\delta_{(f)ij}d_{(f)ij}\sum_{pf=1}\delta_{(f)ij}d_{(f)ij}=\sum_{f=1}p\delta_{ij}(f)d_{ij}(f)\sum_{f=1}p\delta_{ij}(f)$$

dimana  $\delta_{fij}=0$  jika  $x_{if}x_{jf}$  atau  $x_{jf}x_{if}$  adalah hilang (i.e., tidak ada pengukuran dari atribut **f** untuk objek **ii** atau objek **jj**)

- jika  $x_{if}=x_{jf}=0$  dan  $x_{if}\neq x_{jf}$
- atribut **ff** adalah binary asymmetric,

selain itu  $\delta_{fij}=1$

Kontribusi dari atribut **ff** untuk dissimilarity antara **i** dan **j** (yaitu.  $d_{fij}d_{ifj}$ ) dihitung bergantung pada tipenya,

- Jika **ff** adalah numerik,  $d_{fij}=\|x_{if}-x_{jf}\|\max_{h}x_{hf}-\min_{h}x_{hf}$  dan  $d_{ifj}=\|x_{if}-x_{jf}\|\max_{h}x_{hf}-\min_{h}x_{hf}$ , di mana **h** menjalankan semua nilai objek yang tidak hilang untuk atribut **f**
- Jika **ff** adalah nominal atau binary,  $d_{fij}=\begin{cases} 0 & \text{jika } x_{if}=x_{jf} \\ 1 & \text{sebaliknya} \end{cases}$
- Jika **ff** adalah ordinal maka hitung rangking  $r_{if}r_{if}$  dan  $Z_{if}=r_{if}-1M_f-1$   $Z_{if}=r_{if}-1M_f-1$ , dan perlakukan  $Z_{if}Z_{if}$  sebagai numerik.

## Algoritma Hierarchical Clustering

Hierarchical Clustering adalah metode analisis kelompok yang berusaha untuk membangun sebuah hierarki kelompok data.

Strategi pengelompokannya umumnya ada 2 jenis yaitu **Agglomerative (Bottom-Up)** dan **Devisive (Top-Down)**.

### Langkah Algoritma Agglomerative Hierarchical Clustering :

1. Hitung Matrik Jarak antar data.
2. Gabungkan dua kelompok terdekat berdasarkan parameter kedekatan yang ditentukan.
3. Perbarui Matrik Jarak antar data untuk merepresentasikan kedekatan diantara kelompok baru dan kelompok yang masih tersisa.
4. Ulangi langkah 2 dan 3 hingga hanya satu kelompok yang tersisa.

Membentuk Matrik Jarak, misal dengan **Manhattan Distance** :

$$D = \sum_{i=1}^n |b_i - a_i|$$

*Persamaan Manhattan Distance*

atau menggunakan **Euclidian Distance** :

$$D(a, b) = \sqrt{\sum_{i=1}^n (b_i - a_i)^2}$$

*Persamaan Euclidean Distance*

Beberapa metode Pengelompokan **Agglomerative Hierarchical** :

a. **Single Linkage (Jarak Terdekat)**

$$d_{uv} = \min_{single-linkage} \{d_{uv}\}, d_{uv} \in D$$

b. **Complete Linkage (Jarak Terjauh)**

$$d_{uv} = \max_{complete-linkage} \{d_{uv}\}, d_{uv} \in D$$

c. **Average Linkage (Jarak Rata-Rata)**

$$d_{uv} = average \{d_{uv}\}, d_{uv} \in D$$

*average-linkage*

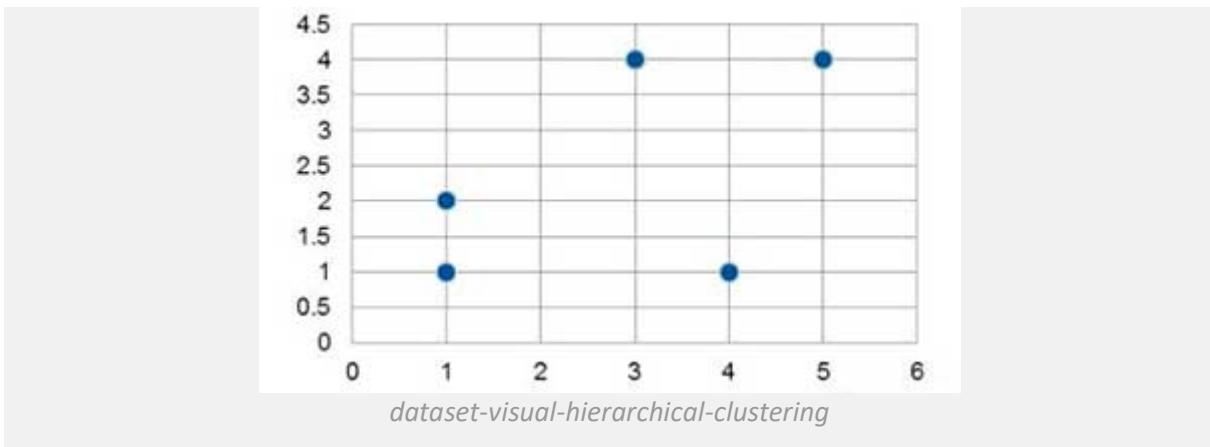
## 1. Contoh Soal Perhitungan

Perhatikan dataset berikut :

Data	Fitur x	Fitur y
1	1	1
2	4	1
3	1	2
4	3	4
5	5	4

*dataset-hierarchical-clustering*

Kelompokkan dataset tersebut dengan menggunakan metode AHC (Single Linkage, Complete Linkage dan Average Linkage) menggunakan jarak Manhattan !



Menghitung Jarak Pada Semua Pasangan dua data :

$$D = \sum_{i=1}^n |b_i - a_i|$$

informatikaologi.com

*Persamaan Manhattan Distance*

$$D_{man} (Data1, Data1) = |1-1| + |1-1| = 0$$

$$D_{man} (Data1, Data2) = |1-4| + |1-1| = 3$$

$$D_{man} (Data1, Data3) = |1-1| + |1-2| = 1$$

$$D_{man} (Data1, Data4) = |1-3| + |1-4| = 5$$

$$D_{man} (Data1, Data5) = |1-5| + |1-4| = 7$$

$$D_{man} (Data2, Data3) = |4-1| + |1-2| = 4$$

$$D_{man} (Data2, Data4) = |4-3| + |1-4| = 4$$

$$D_{man} (Data2, Data5) = |4-5| + |1-4| = 4$$

$$D_{man} (Data3, Data4) = |1-3| + |2-4| = 4$$

$$D_{man} (Data3, Data5) = |1-5| + |2-4| = 6$$

$$D_{man} (Data4, Data5) = |3-5| + |4-4| = 2$$

<b>Dman</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>1</b>	0	3	1	5	7
<b>2</b>	3	0	4	4	4
<b>3</b>	1	4	0	4	6
<b>4</b>	5	4	4	0	2
<b>5</b>	7	4	6	2	0

*Dman-hierarchical-clustering*

### 3. Metode Average Linkage

Dengan memperlakukan data sebagai kelompok, selanjutnya kita pilih jarak dua kelompok yang terkecil.

Dman	1	2	3	4	5
1	0	3	1	5	7
2	3	0	4	4	4
3	1	4	0	4	6
4	5	4	4	0	2
5	7	4	6	2	0

*Dman-hierarchical-clustering*

$$\min(D_{man}) = \min(d_{13}) = 1$$

Terpilih kelompok 1 dan 3, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok (1 dan 3) dengan kelompok lain yang tersisa, yaitu 2, 4 dan 5.

$$d_{(13)2} = \text{average } \{d_{12}, d_{32}\} = \text{average } \{3,4\} = (3+4) / 2 = 3.5$$

$$d_{(13)4} = \text{average } \{d_{14}, d_{34}\} = \text{average } \{5,4\} = (5+4) / 2 = 4.5$$

$$d_{(13)5} = \text{average } \{d_{15}, d_{35}\} = \text{average } \{7,6\} = (7+6) / 2 = 6.5$$

Dengan menghapus baris-baris dan kolom-kolom matrik jarak yang bersesuaian dengan kelompok 1 dan 3, serta menambahkan baris dan kolom untuk kelompok (13).

Dman	(13)	2	4	5
(13)	0	3.5	4.5	6.5
2	3.5	0	4	4
4	4.5	4	0	2
5	6.5	4	2	0

*Dman-hierarchical-clustering-average*

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{45}) = 2$$

Menghitung jarak antar kelompok (4 dan 5) dengan kelompok lain yang tersisa, yaitu (13) dan 2.

$$d_{(45)(13)} = \text{average } \{d_{41}, d_{43}, d_{51}, d_{53}\} = \text{average } \{5,4,7,6\} = (5+4+7+6) / 4 = 5.25$$

$$d_{(45)2} = \text{average } \{d_{42}, d_{52}\} = \text{average } \{4,4\} = (4+4) / 2 = 4$$

Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok 4 dan 5, serta menambahkan baris dan kolom untuk kelompok (45).

Dman	(45)	(13)	2
(45)	0	5.25	4
(13)	5.25	0	3.5
2	4	3.5	0

*Dman-hierarchical-clustering-average-(2)*

Selanjutnya dipilih jarak dua kelompok yang terkecil.

$$\min(D_{man}) = \min(d_{(13)2}) = 3.5$$

Terpilih kelompok (13) dan 2, sehingga kedua kelompok ini digabungkan.

Menghitung jarak antar kelompok ((13) dan 2) dengan kelompok lain yang tersisa, yaitu (45).

$$d_{(45)(13)} = \text{average } \{d_{14}, d_{15}, d_{34}, d_{35}, d_{24}, d_{25}\} = \text{average } \{5, 7, 7, 4, 6, 4\} = (5+7+4+6+4+4) / 6 = 5$$

Menghapus baris dan kolom matrik yang bersesuaian dengan kelompok (45) dan 2, serta menambahkan baris dan kolom untuk kelompok (452).

Dman	(132)	(45)
(132)	0	5
(45)	5	0

*Dman-hierarchical-clustering-average-(3)*

Jadi kelompok (132) dan (45) digabung untuk menjadi kelompok tunggal dari lima data, yaitu kelompok (13245) dengan jarak terdekat 5.

Berikut Dendogram Hasil Metode Average Linkage :

*dendogram-average-linkage*

Rev:

<https://mulaab.github.io/datamining/memahami-data/>



**NAMA : WIDIA ASTUTI**

**NIM : 202420021**

**MATA KULIAH : ADVANCED DATABASE**

Silahkan cari satu tutorial yang membahas tentang pengukuran jarak antar data. Buat ringkasan dari tutorial tersebut dan tulis dalam format ms word, lalu kumpulkan sebelum batas waktu. Sertakan sumber link tutorial yang diringkas.

**Jawab :**

- Salah satu tutorial yang membahas tentang pengukuran jarak antar data yaitu Clustering. Analisis *Cluster* adalah proses dimana akan dilihat pola dari sebuah himpunan data dengan cara mengelompokkan ke dalam beberapa *Cluster*. Tujuannya yaitu untuk menemukan suatu pengelompokan yang sesuai. Sehingga objek pengamatan yang mirip berada pada satu *Cluster*.
- Salah satu metode *clustering* yang berbasis *centroid* yaitu K-Means, dimana metode ini mempresentasikan setiap kelompok dengan suatu pusat kelompok atau *centroid*, serta menghitung jarak setiap data ke setiap *centroid* terdekat. Sehingga *centroid* ini dianggap sebagai nilai pusat baru untuk masing-masing klaster. Analisis kelompok mengacu pada pemartision N objek ke dalam K kelompok (Cluster) berdasarkan nilai rata-rata (means) terdekat.
- Dibawah ini tahapan K-Means Clustering :
  1. Tentukan berapa banyak jumlah k (cluster)
  2. Secara acak tentukan record yang menjadi lokasi pusat cluster.
  3. Temukan pusat cluster terdekat untuk setiap record. Adapun persamaan yang sering digunakan dalam pemecahan masalah dalam menentukan jarak terdekat adalah persamaan Euclidean berikut :

$$d_{Euclidean}(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$$

Dimana  $x=x_1,x_2,x_3,\dots,x_m$  dan  $y=y_1,y_2,y_3,\dots,y_m$ , sementara m menyatakan banyaknya nilai atribut dari 2 buah record.

4. Tentukan cluster terdekat untuk setiap data dengan membandingkan nilai jarak terdekat, lalu perbarui nilai pusat clusternya.

$$ClusterCenter = \sum \frac{a_i}{n}$$

5. Ulangi langkah 3 sampai 5 hingga tidak ada record yang berpindah cluster atau convergen.

**SUMBER :** <https://www.alfasoleh.com/2019/11/k-means-clustering-contoh-sederhana.html?m=1>

[https://www.youtube.com/watch?v=\\_BUoC7P7mnY&t=406s](https://www.youtube.com/watch?v=_BUoC7P7mnY&t=406s)

[https://www.youtube.com/watch?v=5o\\_JQ6AHA0Y&t=365s](https://www.youtube.com/watch?v=5o_JQ6AHA0Y&t=365s)

### CONTOH :

Dibawah ini terdapat data 8 record, jika data tsb dikelompokkan kedalam dua kluster. Tentukan pusat dan anggota dari masing-masing klaster tsb.

Data ke-	Atribut	
	V1	V2
1	1	3
2	3	3
3	4	3
4	5	3
5	1	2
6	4	2
7	1	1
8	2	1

### Jawab :

Terdapat 8 record yang akan menjadi dataset kemudian dataset tersebut akan kita gunakan dalam membantu memahami penerapan algoritma k-means clustering.

Langkah-langkahnya :

1. Kita tentukan 2 cluster (kelompok)
2. Tetapkan 2 record dari dataset sebagai titik pusat cluster.  
 $M1 : \{1,1\} \rightarrow$  Titik pusat Cluster pertama (C1)  
 $M2 : \{2,1\} \rightarrow$  Titik pusat Cluster kedua (C2)
3. tentukan pusat cluster terdekat untuk setiap record dari dataset, kita akan menggunakan persamaan *Euclidean* untuk menentukan jarak setiap record dengan pusat cluster.

Data ke 1 : {1,3}

$\begin{aligned} d_{\text{Euclidean}}(M1) &= \sqrt{(1-1)^2 + (3-1)^2} \\ &= \sqrt{(0)^2 + (2)^2} \\ &= \sqrt{0+4} \\ &= \sqrt{4} \\ &= 2 \end{aligned}$	$\begin{aligned} d_{\text{Euclidean}}(M2) &= \sqrt{(1-2)^2 + (3-1)^2} \\ &= \sqrt{(-1)^2 + (2)^2} \\ &= \sqrt{1+4} \\ &= \sqrt{5} \\ &= 2.24 \end{aligned}$
---------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------

Data ke 2 : {3,3}

$\begin{aligned} d_{\text{Euclidean}}(M1) &= \sqrt{(3-1)^2 + (3-1)^2} \\ &= \sqrt{(2)^2 + (2)^2} \\ &= \sqrt{4+4} \\ &= \sqrt{8} \\ &= 2.83 \end{aligned}$	$\begin{aligned} d_{\text{Euclidean}}(M2) &= \sqrt{(3-2)^2 + (3-1)^2} \\ &= \sqrt{(1)^2 + (2)^2} \\ &= \sqrt{1+4} \\ &= \sqrt{5} \\ &= 2.24 \end{aligned}$
------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------

Dan dihitung seterusnya hingga data paling akhir (data ke 8), sehingga diperoleh rekap hasil perhitungan jarak terdekat ke setiap cluster sebagai berikut :

Data	V1	V2	C1	C2
1	1	3	2.00	2.24
2	3	3	2.83	2.24
3	4	3	3.61	2.83
4	5	3	4.47	3.61
5	1	2	1.00	1.41
6	4	2	3.16	2.24
7	1	1	0.00	1.00
8	2	1	1.00	0.00

4. tentukan cluster(kelompok) setiap record dan perbaharui titik pusat cluster.

Data	V1	V2	C1	C2	Cluster
1	1	3	2.00	2.24	C1
2	3	3	2.83	2.24	C2
3	4	3	3.61	2.83	C2
4	5	3	4.47	3.61	C2
5	1	2	1.00	1.41	C1
6	4	2	3.16	2.24	C2
7	1	1	0.00	1.00	C1
8	2	1	1.00	0.00	C2

Setelah cluster(kelompok) untuk setiap record ditentukan seperti pada tabel di atas, tugas kita sekarang adalah memperbaharui nilai titik pusat cluster, yang mana sebelumnya titik pusat cluster kita adalah M1{1,1} dan M2{2,1}. Untuk meng-update nilai titik pusat cluster, kita dapat menggunakan persamaan cluster center sebagai berikut :

Cluster 1 (C1) :

$$\begin{aligned} \text{ClusterCenter}(M1(x)) &= \frac{x_1 + x_5 + x_7}{3} \\ &= \frac{1 + 1 + 1}{3} \\ &= \frac{3}{3} \\ &= 1 \end{aligned}$$

$$\begin{aligned} \text{ClusterCenter}(M1(y)) &= \frac{y_1 + y_5 + y_7}{3} \\ &= \frac{3 + 2 + 1}{3} \\ &= \frac{6}{3} \\ &= 2 \end{aligned}$$

Cluster 2 (C2) :

$\begin{aligned} ClusterCenter(M1(x)) \\ = \frac{x_2 + x_3 + x_4 + x_6 + x_8}{5} \\ = \frac{3 + 4 + 5 + 4 + 2}{5} \\ = \frac{18}{5} \\ = 3.6 \end{aligned}$	$\begin{aligned} ClusterCenter(M1(y)) \\ = \frac{y_2 + y_3 + y_4 + y_6 + y_8}{5} \\ = \frac{3 + 3 + 3 + 2 + 1}{5} \\ = \frac{12}{5} \\ = 2.4 \end{aligned}$
-------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------

Maka titik pusat cluster terbaru adalah sebagai berikut :

UPDATE Pusat Cluster
Pusat Cluster
M1                    1            2
M2                    3.6        2.4

5. Setelah didapat nilai update titik pusat cluster, selanjutnya kita akan mengulangi langkah ketiga, guna menentukan kembali kelompok tiap data. Untuk menguji apakah terjadi perpindahan kelompok pada data tersebut. setelah di hitung kembali didapatkan hasil nilai terdekat tiap kelompok sebagai berikut.

ITERASI 1					
	Data	V1	V2	C1	C2
	1	1	3	1.00	2.67
	2	3	3	2.24	0.85
	3	4	3	3.16	0.72
	4	5	3	4.12	1.52
	5	1	2	0.00	2.63
	6	4	2	3.00	0.57
	7	1	1	1.00	2.95
	8	2	1	1.41	2.13

sementara untuk penentuan kelompok dapat dilihat pada tabel di bawah ini

ITERASI 1							
	Data	V1	V2	C1	C2	Cluster Sebelumnya	Cluster Baru
	1	1	3	1.00	2.67	C1	C1
	2	3	3	2.24	0.85	C2	C2
	3	4	3	3.16	0.72	C2	C2
	4	5	3	4.12	1.52	C2	C2
	5	1	2	0.00	2.63	C1	C1
	6	4	2	3.00	0.57	C2	C2
	7	1	1	1.00	2.95	C1	C1
	8	2	1	1.41	2.13	C2	C1

Setelah dilakukan penentuan cluster yang baru, sesuai dengan titik pusat cluster yang diupdate sebelumnya, ditemukan adanya 1 data yang berpindah kelompok, yang mengakibatkan pengelompokan pada iterasi pertama ini belum konvergen, sehingga harus dilakukan penentuan kelompok kembali pada iterasi kedua dengan nilai titik pusat cluster yang harus diupdate ulang.

UPDATE TITIK PUSAT CLUSTER		
M1	1.25	1.75
M2	4	2.75

Adapun hasil pengelompokan setelah melakukan ulang langkah 3 dan langkah 4 adalah sebagai berikut :

ITERASI 2		V1	V2	C1	C2	Cluster Sebelumnya	Cluster Iterasi 1	Cluster Baru
	1	1	3	1.27	3.01	C1	C1	C1
	2	3	3	2.15	1.03	C2	C2	C2
	3	4	3	3.02	0.25	C2	C2	C2
	4	5	3	3.95	1.03	C2	C2	C2
	5	1	2	0.35	3.09	C1	C1	C1
	6	4	2	2.76	0.75	C2	C2	C2
	7	1	1	0.79	3.47	C1	C1	C1
	8	2	1	1.06	2.66	C2	C1	C1

Pada iterasi 2, tidak terjadi perpindahan kelompok pada setiap data, dengan ini maka pengelompokan sudah dinyatakan konvergen atau sudah dianggap optimal.

# TUGAS ADVANCED DATABASE



Dibuat Oleh

**Aan Novrianto**

Dosen Pengampu

**TRI BASUKI KURNIAWAN, S.KOM, M.Eng.Ph.D**

Program Pasca Sarjana

Universitas Binadarma Palembang

## PENGUKURAN DATA

*Clustering* merupakan aktivitas (*task*) yang bertujuan mengelompokkan data yang memiliki kemiripan antara satu data dengan data lainnya ke dalam klaster atau kelompok sehingga data dalam satu klaster memiliki tingkat kemiripan (*similarity*) yang maksimum dan data antar klaster memiliki kemiripan yang minimum. *Clustering* juga dapat diartikan metode segmentasi data yang diimplementasikan dalam beberapa bidang, diantaranya marketing, analisa masalah bisnis segmentasi pasar dan prediksi, pola dalam bidang computer vision, zonasi wilayah hingga identifikasi obyek dan pengolahan citra. Analisis klaster bertujuan menemukan kelompok objek sedemikian rupa sehingga objek-objek dalam grup akan sama (atau terkait) satu sama lain dan berbeda dari (atau tidak terkait) objek-objek dalam grup lain [1]. Ada sejumlah algoritma yang dapat digunakan untuk pengelompokan. Secara umum, K-Means merupakan algoritma heuristik yang memisahkan kumpulan data ke dalam klaster K dengan meminimalkan jumlah jarak kuadrat di setiap klaster. Pada penelitian sebelumnya [2], penerapan algoritma K-Means dasar telah dilakukan dengan menggunakan metode pengukuran jarak *Euclidean* (*Euclidean Distance*). Pada penelitian ini, diimplementasikan pengukuran jarak menggunakan metode *Manhattan* dan *Minkowski* pada algoritma K-Means berbasis *Chi-Square*. Selain itu, juga dilakukan perbandingan tingkat akurasi dari masing-masing metode untuk mengetahui metode terbaik.

Pengukuran jarak memegang peran yang sangat penting dalam menentukan kemiripan atau keteraturan di antara data dan item. hal ini dilakukan untuk mengetahui, dengan cara seperti apa data dikatakan saling terkait, mirip, tidak mirip, dan metode pengukuran jarak seperti apa yang diperlukan untuk membandingkannya [10]. Pada proses *clustering*, tahapan menentukan atau mendeskripsikan nilai kuantitatif dari tingkat kemiripan atau ketidakmiripan data (*proximity measure*) memiliki peranan sangat penting, sehingga perlu dilakukannya perbandingan beberapa metode yang sering digunakan, yaitu jarak *euclidean*, *manhattan*, dan *minkowski*

### A. Euclidean Distance

*Euclidean distance* merupakan salah satu metode perhitungan jarak yang digunakan untuk mengukur jarak dari 2 (dua) buah titik dalam *Euclidean space* (meliputi bidang *euclidean* dua dimensi, tiga dimensi, atau bahkan lebih). Untuk mengukur tingkat kemiripan data dengan rumus *euclidean distance* digunakan rumus berikut

$$d(x, y) = |x - y| = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

dimana,

d = jarak antara x dan y

x = data pusat klaster

y = data pada atribut

i = setiap data

n = jumlah data,

$x_i$  = data pada pusat klaster ke i

$y_i$  = data pada setiap data ke i

### B. Manhattan Distance

*Manhattan distance* digunakan untuk menghitung perbedaan absolut (mutlak) antara koordinat sepasang objek. Rumus yang digunakan adalah sebagai berikut:

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

dimana,

$d$  = jarak antara  $x$  dan  $y$

$x$  = data pusat klaster

$y$  = data pada atribut

$i$  = setiap data

$n$  = jumlah data,

$x_i$  = data pada pusat klaster ke  $i$

$y_i$  = data pada setiap data ke  $i$

### C. Minkowski Distance

*Minkowski distance* merupakan sebuah metrik dalam ruang vektor di mana suatu norma didefinisikan (*normed vector space*) sekaligus dianggap sebagai generalisasi dari *Euclidean distance* dan *Manhattan distance*. Dalam pengukuran jarak objek menggunakan *minkowski distance* biasanya digunakan nilai  $p$  adalah 1 atau 2. Berikut rumus yang digunakan menghitung jarak dalam metode ini.

$$d(x, y) = \left( \sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}$$

dimana,

$d$  = jarak antara  $x$  dan  $y$

$x$  = data pusat klaster

$y$  = data pada atribut

$i$  = setiap data

$n$  = jumlah data,

$x_i$  = data pada pusat klaster ke  $i$

$y_i$  = data pada setiap data ke  $i$

$p$  = power

Sumber :

Jurnal Informatika: Jurnal Pengembangan IT (JPIT), Vol.04, No.01, Januari 2019

ISSN: 2477-5126

DOI: 10.30591/jpit.v4i1.1253

[https://www.researchgate.net/publication/332758314\\_Perbandingan\\_Akurasi\\_Euclidean\\_Distance\\_Minkowski\\_Distance\\_dan\\_Manhattan\\_Distance\\_pada\\_Algoritma\\_K-Means\\_Clustering\\_berbasis\\_Chi-Square](https://www.researchgate.net/publication/332758314_Perbandingan_Akurasi_Euclidean_Distance_Minkowski_Distance_dan_Manhattan_Distance_pada_Algoritma_K-Means_Clustering_berbasis_Chi-Square)

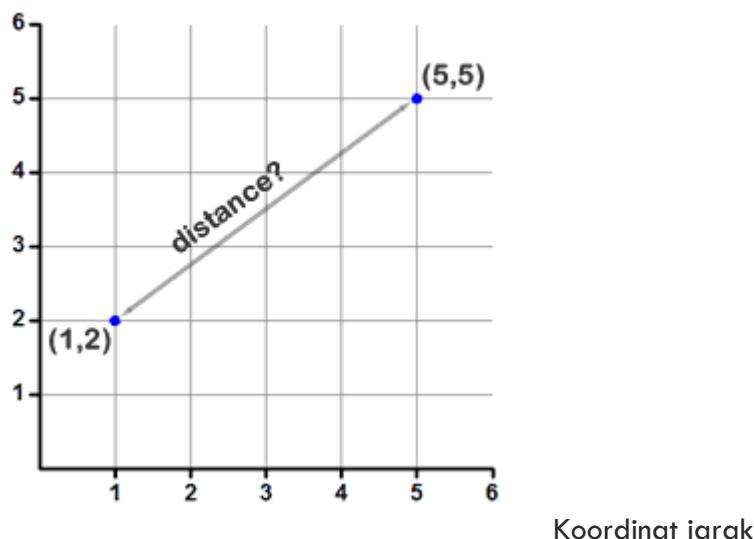
## Contoh Perhitungan Teori Euclidean Distance

Euclidean distance adalah perhitungan jarak dari 2 buah titik dalam Euclidean space. Euclidean space diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini berkaitan dengan Teorema Phytagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tapi juga sederhana jika diterapkan pada dimensi yang lebih tinggi.

### Pada 1 dimensi

Semisal ingin menghitung jarak Euclidean 1 dimensi. Titik pertama adalah 4, titik kedua adalah -10. Caranya adalah kurangkan -10 dengan 4. sehingga menghasilkan -14. Cari nilai absolut dari nilai -14 dengan cara mem pangkatkannya sehingga mendapat nilai 196. Kemudian diakarkan sehingga mendapatkan nilai 14. Sehingga jarak euclidean dari 2 titik tersebut adalah 14.

### Pada 2 dimensi



Caranya hampir sama. Misalkan titik pertama mempunyai kordinat (1,2). Titik kedua ada di kordinat (5,5). Caranya adalah kurangkan setiap kordinat titik kedua dengan titik yang pertama. Yaitu, (5-1,5-2) sehingga menjadi (4,3). Kemudian pangkatkan masing-masing sehingga memperoleh (16,9). Kemudian tambahkan semuanya sehingga memperoleh nilai  $16+9 = 25$ . Hasil ini kemudian diakarkan menjadi 5. Sehingga jarak euclideanya adalah 5.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Rumus Euclid

Sehingga dari Formula diatas kita dapat implementasi menjadi :

$$Jarak = \sqrt{(Lat_1 - Lat_2)^2 + (Long_1 - Long_2)^2}$$

Rumus

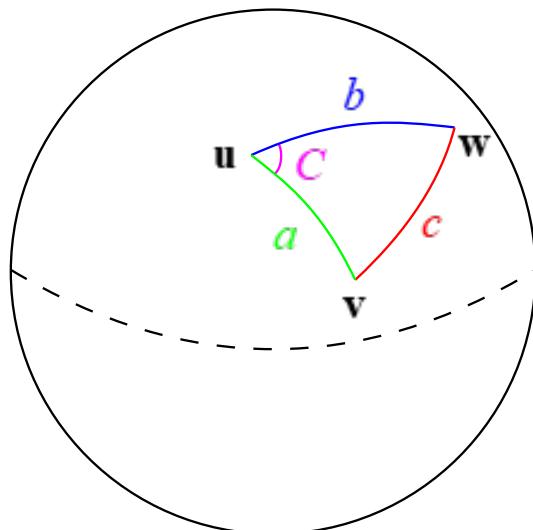
#### Jarak Euclide

Hasil perhitungan (Jarak) diatas masih dalam satuan decimal degree (sesuai dengan format longlat yang dipakai) sehingga untuk menyesuaikannya perlu dikalikan dengan **111.319 km** (1 derajat bumi = 111.319 km)

---

#### Teori Haversine Formula

Teorema Haversine Formula adalah sebuah persamaan yang penting dalam bidang navigasi, untuk mencari jarak busur antara dua titik pada bola dari longitude dan latitude. Ini merupakan bentuk persamaan khusus dari trigonometri bola, law of haversines, mencari hubungan sisi dan sudut pada segitiga dalam bidang bola.

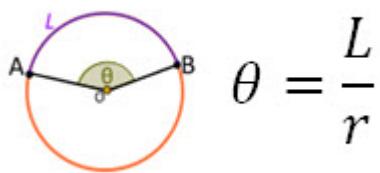


Ilustrasi Spherical law of cosines

$$\cos(c) = \cos(a) \cos(b) + \sin(a) \sin(b) \cos(C)$$

Spherical law of cosines

Dimana a,b,c ialah jarak yang bersatuan radian/sudut karena berada dalam bidang bola, yang bisa kita korelasikan dengan persamaan busur dibawah ini :



Rumus Busur

Kemudian kita implementasikan persamaan harvesin dibawah ini :

$$\text{haversin}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos(\theta)}{2}$$

Harvesin Formula

Sehingga dari Formula diatas kita dapat implementasi menjadi :

$$\text{Jarak} = 2r \cdot \arcsin \left\{ \sqrt{\sin^2\left(\frac{Lat_1 - Lat_2}{2}\right) + \cos(Lat_1) \cdot \cos(Lat_2) \cdot \sin^2\left(\frac{Long_1 - Long_2}{2}\right)} \right\}$$


---

Rumus Jarak Harvesin

### Penggunaan Teorema Euclid dan Teorema Harvesine

Pertama kita siapkan bahan uji sebagai berikut :

#### 1. Titik Pertama : Gedung Sate (Bandung)

Long : 107.618633

Lat : -6.901361

#### 2. Titik Kedua: Mesjid Raya Lembang (KBB)

Long : 107.618279

Lat : -6.811771

Jarak aktual yang diperoleh dari Gmap ialah 10.05 Km.



Jarak Lurus Bandung Lembang

Baiklah, bahan sudah ada tinggal kita implementasi dengan rumus yang sudah ada. Saya akan sertakan rumus excelnya supaya lebih mudah membuktikannya.

**Dengan Teorema Euclid :**

$$Jarak = \sqrt{(Lat_1 - Lat_2)^2 + (Long_1 - Long_2)^2}$$

Rumus

Jarak Euclidean  
=  $((\text{SQRT}((B6-B7)^2+(C6-C7)^2)*111.319))$

Hasilnya diperoleh Jarak = 9.97 Km

Dengan Teorema Harvesine:

$$Jarak = 2r \cdot \arcsin \left\{ \sqrt{\sin^2\left(\frac{Lat_1 - Lat_2}{2}\right) + \cos(Lat_1) \cdot \cos(Lat_2) \cdot \sin^2\left(\frac{Long_1 - Long_2}{2}\right)} \right\}$$

Rumus Jarak Harvesin

= $(6371.1 * ((2 * \text{ASIN}(\text{SQRT}((\text{SIN}((\text{RADIANS}(B7) - \text{RADIANS}(B6)) / 2)^2) + \text{COS}(\text{RADIANS}(B7)) * \text{COS}(\text{RADIANS}(B6)) * (\text{SIN}((\text{RADIANS}(C7) - \text{RADIANS}(C6)) / 2)^2))))$

Hasilnya diperoleh Jarak = 9.96 Km

Hasil perhitungan Excel :

Perhitungan Jarak melalui koordinat			
	Latitude	Longitude	
Gedung Sate	-6.901361	107.618633	
Mesjid Lembang	-6.811771	107.618279	
Euclidean =	9.97	Km	= $((\text{SQRT}((C5-C6)^2+(D5-D6)^2)*111.319))$
Harvesine =	9.96	Km	= $(6371.1 * ((2 * \text{ASIN}(\text{SQRT}((\text{SIN}((\text{RADIANS}(B7) - \text{RADIANS}(B6)) / 2)^2) + \text{COS}(\text{RADIANS}(B7)) * \text{COS}(\text{RADIANS}(B6)) * (\text{SIN}((\text{RADIANS}(C7) - \text{RADIANS}(C6)) / 2)^2))))$

Hasil Excel

Dari kedua teorema diatas hasil yang diperoleh tidak berbeda jauh meskipun ada selisih 0.09 dengan hasil Gmap. sehingga cukup ampuh untuk menghitung jarak2 pendek yang kedepannya bisa diakumulasi untuk memperoleh jarak sesuai jalur yang dilalui.

Sumber :

<https://blogs.itb.ac.id/anugraha/2014/09/10/teori-pengukuran-jarak/>

### **Cara mengukur jarak menggunakan Manhattan Distance (city block distance)**

Artikel singkat kali ini akan membahas cara lain mengukur jarak selain Euclidean yang sudah pernah juga dituliskan. Disebut Manhattan ini berdasar pada kota Manhattan yang tersusun menjadi blok-blok. Sehingga sering juga disebut city block distance, juga sering disebut sebagai absolute value distance atau boxcar distance.

Sebagai ilustrasi, semisal kita berjalan dari lokasi A menuju utara 3 meter, kemudian belok ke timur 4 meter. Berapakah jarak kita yang sekarang dengan posisi titik A tadi. **City Block distance adalah panjang jalan yang sudah kita tempuh dari B ke A.**

Sebagai contoh, kita coba perhitungan berikut ini :

Jika kita ambil contoh 2 titik (objek) dengan koordinat dari object A (0,3,4,5). Object B (7,6,3,-1)

Maka City Block distance antara titik A dan titik B adalah :

$$d = |0-7| + |3-6| + |4-3| + |5+1| = 7+3+1+6=17$$

Sumber :

<http://itmbali.blogspot.com/2012/11/cara-mengukur-jarak-menggunakan.html>



Berikut adalah contoh perhitungan manual mengenai algoritma k-means antara lain :

1. Dataset

Tabel 1 merupakan tabel dataset dari 15 mahasiswa yang memprogramkan mata kuliah Data mining. Dari 15 mahasiswa tersebut akan dikelompokkan menjadi 3 bagian yaitu kelompok pintar, sedang dan kurang.

Tabel 1. Dataset

NO	NAMA MAHASISWA	UTS	TUGAS	UAS
1	Roy	89	90	75
2	Sintia	90	71	95
3	Iqbal	70	75	80
4	Dilan	45	65	59
5	Ratna	65	75	53
6	Merry	80	70	75
7	Rudi	90	85	81
8	Hafiz	70	70	73
9	Gede	96	93	85
10	Christian	60	55	48
11	Justin	45	60	58
12	Jesika	60	70	72
13	Ayu	85	90	88
14	Siska	52	68	55
15	Reitama	40	60	70

2. Setelah menentukan dataset, maka perlu menentukan jumlah cluster yang akan dibentuk. Adapun cluster yang akan dibentuk antara lain :

- a. Cluster 1 (C1) = Pintar
  - b. Cluster 2 (C2) = Sedang
  - c. Cluster 3 (C3) = Kurang
3. Tetapkan C pusat cluster awal secara random

Dari dataset diatas terpilih 3 cluster pusat diantaranya :

Kluster 1	96	93	85
Kluster 2	70	75	80
Kluster 3	60	55	48

4. Alokasikan semua data/obyek ke dalam cluster terdekat. Berikut hasil dari alokasi data ke jarak cluster.

Adapun hasil dari jarak ke cluster diperoleh dari perhitungan dengan rumus :

(lakukan perhitungan tersebut sampai data ke 15)

Setelah melakukan perhitungan maka didapat hasil seperti berikut ini :

No	Nama Mahasiswa	Jarak Ke Cluster			Hasil
		C1	C2	C3	
1.	Roy	12,56980509	24,71841419	52,86775955	1
2.	Sintia	24,8997992	25,3179778	58,00862005	1
3.	Iqbal	32,01562119	0	39,03844259	2
4.	Dilan	63,72597587	34,14674216	21,11871208	3
5.	Ratna	48,05205511	27,45906044	21,21320344	3
6.	Merry	29,74894956	12,24744871	36,79673899	2

7.	Rudi	10,77032961	22,38302929	53,7494186	1
8.	Hafiz	36,72873534	8,602325267	30,82207001	2
9.	Gede	0	32,01562119	64,10148204	1
10.	Christian	64,10148204	39,03844259	0	3
11.	Justin	66,47555942	36,52396474	18,70828693	3
12.	Jesika	44,65422712	13,74772708	28,3019434	2
13.	Ayu	11,78982612	22,6715681	58,73670062	1
14.	Siska	58,83026432	31,591138	16,79285562	3
15.	Reitama	66,70832032	35	30,14962686	3

1. Tentukan kembali titik pusat cluster yang baru berdasarkan rata-rata

Cluster baru tersebut didapat dari rumus = nilai hasil /banyak hasil

$$\text{Kluster 1 (UTS)} = (89+90+90+90+85)/5=90$$

$$\text{Kluster 1 (Tugas)} = (90+71+85+93+90)/5=85,8$$

$$\text{Kluster 1 (UAS)} = (75+95+81+85+88)/5=84,8$$

Lakukan, perhitungan tersebut untuk kluster 2 dan 3, sehingga didapat nilai cluster baru antara lain :

Kluster 1	90	85,8	84,8
Kluster 2	70	71,25	75
Kluster 3	51,16666667	63,83333333	57,16666667

2. Lakukan kembali langkah 4 hingga titik pusat dari setiap cluster tidak berubah

Berikut hasil yang didapat sesuai dengan langkah ke 4

No	Nama Mahasiswa	Jarak Ke Cluster			Hasil
		C1	C2	C3	
1.	Roy	10,70887482	26,69386634	49,33643008	1
2.	Sintia	17,97442628	28,28537608	54,68775	1
3.	Iqbal	23,23101375	6,25	31,63463292	2
4.	Dilan	55,88631317	30,33253204	6,538348415	3
5.	Ratna	41,86740976	22,87055968	18,25970062	3
6.	Merry	21,1111345	10,07782219	34,45891273	2
7.	Rudi	3,883297568	25,00124997	50,2402561	1
8.	Hafiz	28,08700767	2,358495283	25,3656592	2
9.	Gede	9,374433316	35,34207832	60,29441655	1
10.	Christian	56,59399261	33,06149573	15,49462272	3
11.	Justin	58,38561467	32,25775101	7,30867065	3
12.	Jesika	36,24196463	10,51487042	18,33257574	2
13.	Ayu	7,271863585	27,30499039	52,72649555	1
14.	Siska	51,46727115	27,10281351	4,769696007	3
15.	Reitama	58,17800272	32,42780443	17,43798536	3

Hasil dari tahapan yang pertama dan kedua tidak berubah, maka hasil sudah sesuai dengan pengelompokan kluster. Berikut adalah hasil dari pengelompokan tersebut

No	Nama Mahasiswa	UTS	Tugas	UAS	Kelompok
1.	Roy	89	90	75	Pintar
2.	Sintia	90	71	95	Pintar

No	Nama Mahasiswa	UTS	Tugas	UAS	Kelompok
3.	Iqbal	70	75	80	Sedang
4.	Dilan	45	65	59	Kurang
5.	Ratna	65	75	53	Kurang
6.	Merry	80	70	75	Sedang
7.	Rudi	90	85	81	Pintar
8.	Hafiz	70	70	73	Sedang
9.	Gede	96	93	85	Pintar
10.	Christian	60	55	48	Kurang
11.	Justin	45	60	58	Kurang
12.	Jesika	60	70	72	Sedang
13.	Ayu	85	90	88	Pintar
14.	Siska	52	68	55	Kurang
15.	Reitama	40	60	70	Kurang

**ADVANCE DATABASE**  
**“RINGKASAN TUTORIAL PENGUKURAN JARAK ANTAR DATA”**

**TUGAS 1**

**OLEH :**  
**202420017 Ainun Hilaliyyah**

**DOSEN PENGAMPUH :**  
**Bpk. Tri Basuki Kurniawan, S.Kom., M.eng.,Ph.D**

**Program Studi IT Infrastructure**



**UNIVERSITAS BINA DARMA  
PALEMBANG  
2020**

## PENGUKURAN JARAK ANTAR DATA

Salah satu tahap menentukan kesamaan atau kemiripan data bisa menggunakan metode pengukuran jarak. Karena jarak merupakan aspek penting dalam pengembangan metode pengelompokan. Sebelum pengelompokan data di proses, harus menentukan ukuran jarak kedekatan antar data.

Adapun metode yang digunakan adalah :

1. Euclidean Distance
2. Manhattan Distance/ Minkowski Distance
3. Canberra Distance
4. Mahalanobis Distance

Dalam banyak penelitian, Euclidean Distance adalah metode pengukuran jarak yang sering digunakan. Euclidean distance adalah perhitungan jarak dari 2 buah titik dalam Euclidean Space, untuk mempelajari hubungan antara sudut dan jarak. Berikut adalah contoh perhitungan jarak antar data menggunakan metode Euclidean Distance;

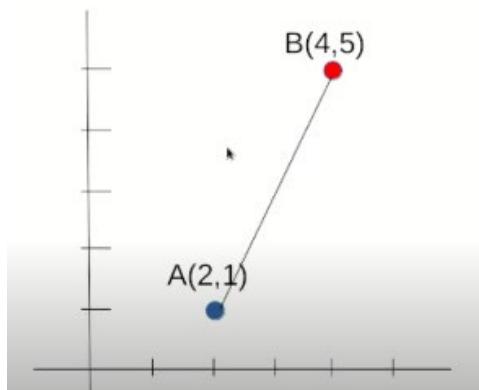
### 1. Contoh Data 2 Dimensi

Diketahui:

	$X_1$	$X_2$
A	2	1
B	4	5

Gambar 1.

Ilustrasi



Untuk mendapatkan jarak antar data tersebut, dapat dihitung menggunakan rumus Euclidean Distance.

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

$x = a, y = b$   
 $i = \text{dimensi atribut}$   
 $n = \text{jumlah atribut}$

Penyelesaian :

$$\begin{aligned} d(A, B) &= \sqrt{(2 - 4)^2 + (1 - 5)^2} \\ &= (-2)^2 + (-4)^2 = 20 \\ &= \sqrt{20} = 4.47 \end{aligned}$$

Jadi jarak garis lurus yaitu 4.47.

## 2. Contoh Data 3 Dimensi

	X <sub>1</sub>	X <sub>2</sub>	X <sub>3</sub>
A	2	1	3
B	4	5	2

Gambar 2.

Penyelesaian :

$$\begin{aligned} d(A, B) &= \sqrt{(2 - 4)^2 + (1 - 5)^2 + (3 - 2)^2} \\ &= (-2)^2 + (-4)^2 + (1)^2 = 21 \\ &= \sqrt{21} = 4.58 \end{aligned}$$

Sumber: [https://www.youtube.com/watch?v=PEiw5e\\_E6A8](https://www.youtube.com/watch?v=PEiw5e_E6A8)

**Nama** : Andry Meylani  
**NIM** : 202420009  
**Source** : <https://pemrogramanmatlab.com/2017/07/26/pencocokan-citra/#more-5343>

## Pencocokan Citra dengan Mengukur Jarak Euclidean

Pencocokan citra (*image matching*) merupakan salah satu bagian dari pengolahan citra yang dilakukan untuk mencari citra lain yang sejenis atau memiliki kemiripan. Salah satu parameter yang merepresentasikan tingkat kemiripan antara dua buah citra adalah jarak euclidean. Semakin kecil jarak euclidean antara dua buah citra maka akan semakin mirip kedua citra tersebut. Persamaan untuk menghitung jarak euclidean adalah sebagai berikut:

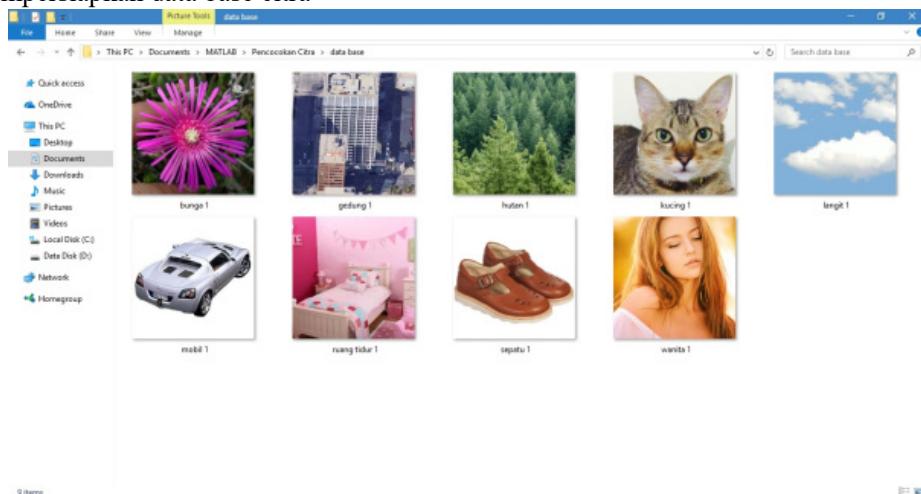
$$\text{dist}(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Jarak euclidean dapat dihitung berdasarkan ciri khusus yang dimiliki oleh suatu citra. Ciri tersebut di antaranya adalah ciri warna, ciri tekstur, ciri bentuk, ciri geometri, dan ciri ukuran.

Berikut ini merupakan contoh aplikasi sistem pencocokan citra menggunakan jarak *euclidean* berdasarkan pada ciri warna. Ciri warna dihitung pada ruang warna HSV yang terdiri dari komponen *Hue*, *Saturation*, dan *Value*. Sistem pencocokan citra diimplementasikan dalam bentuk tampilan GUI menggunakan bahasa pemrograman MATLAB.

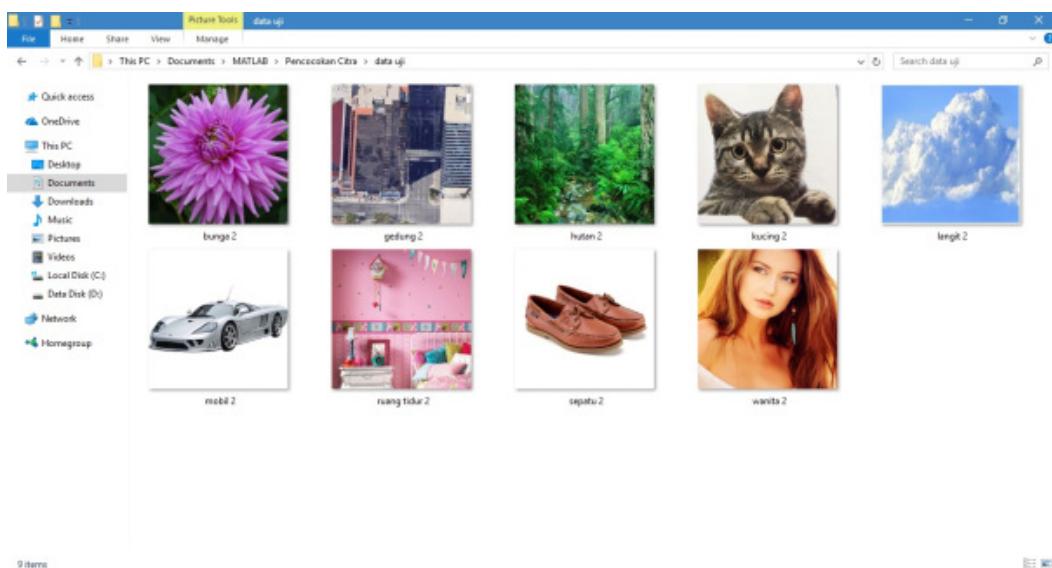
Langkah-langkah pemrogramannya adalah sebagai berikut:

1. Mempersiapkan data base citra



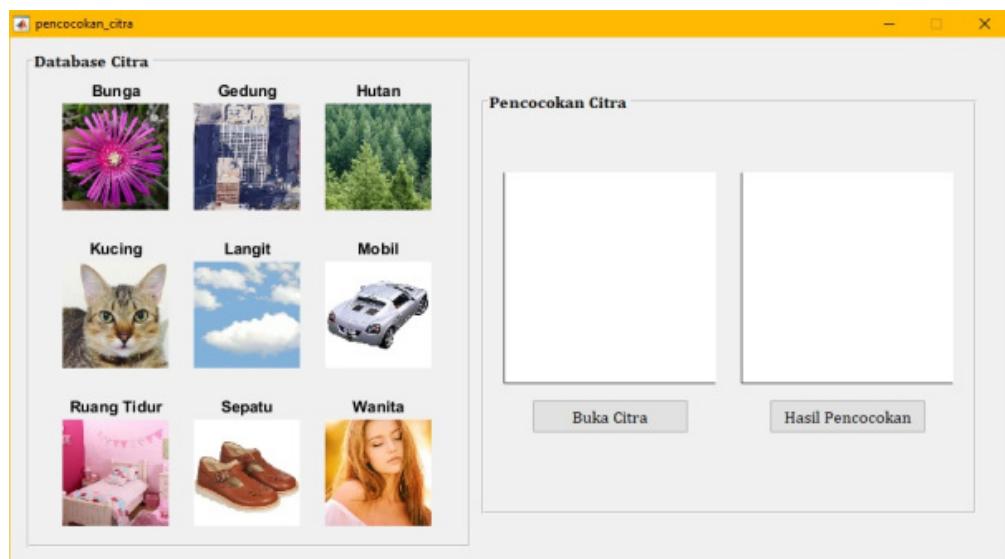
Data base menggunakan sembilan buah citra yaitu berupa citra bunga, citra gedung, citra hutan, citra kucing, citra langit, citra mobil, citra ruang tidur, citra sepatu, dan citra wanita.

## 2. Mempersiapkan data uji citra



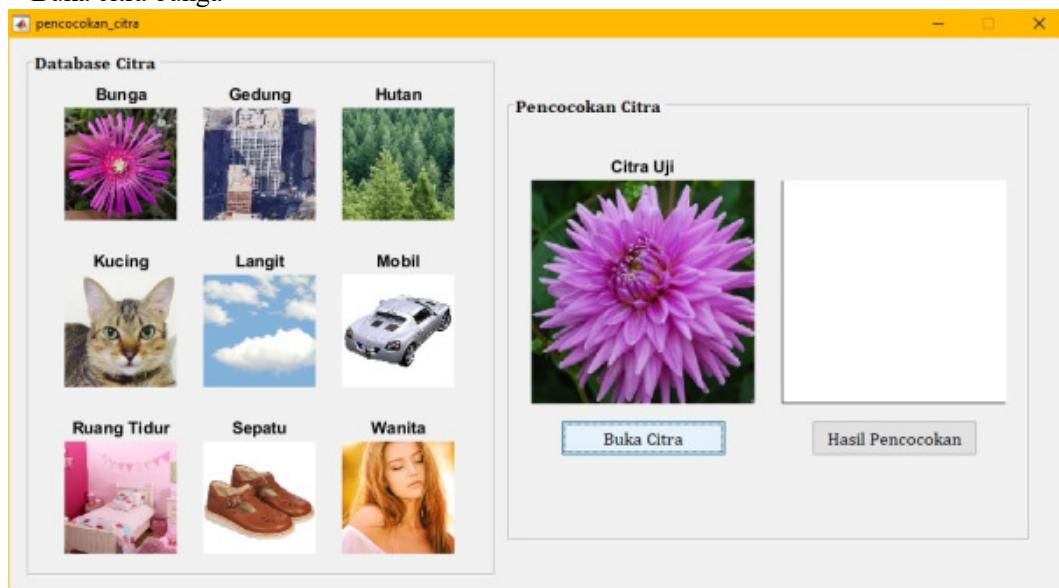
Sama seperti pada data base citra, pada data uji citra juga menggunakan citra yang sejenis.

## 3. Membuka tampilan GUI awal

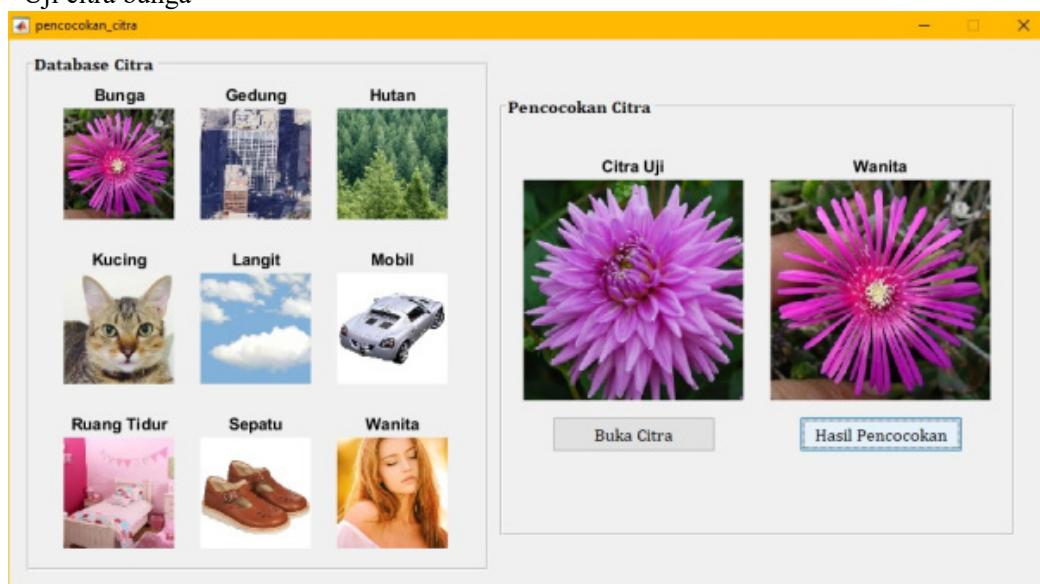


Pada tampilan GUI awal ditunjukkan database yang berjumlah sembilan buah citra. Kesembilan database citra tersebut masing-masing akan dihitung jaraknya dengan sebuah citra uji. Dari kesembilan jarak yang diperoleh, penentuan pasangan citra yang cocok atau sejenis dengan citra uji adalah citra pada database yang memiliki jarak euclidean terdekat.

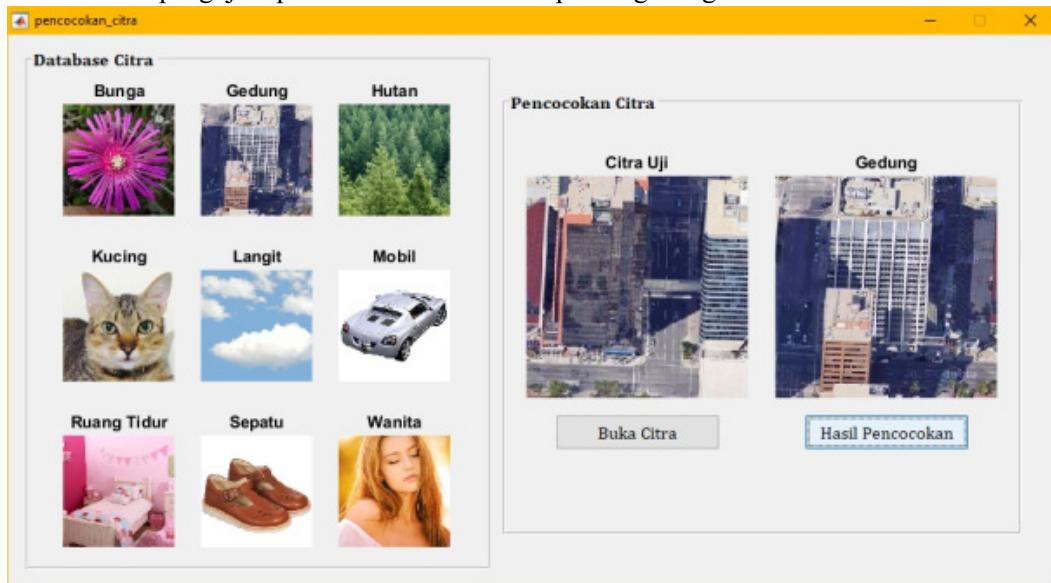
4. Membuka salah satu citra pada data uji yaitu citra bunga dan melakukan pengujian pencocokan citra
- Buka citra bunga



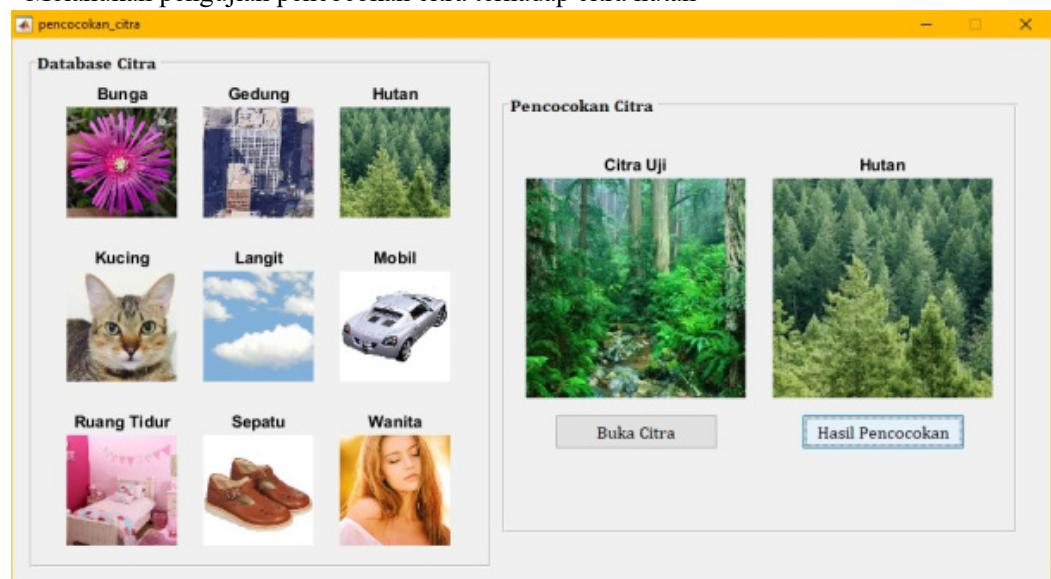
Uji citra bunga



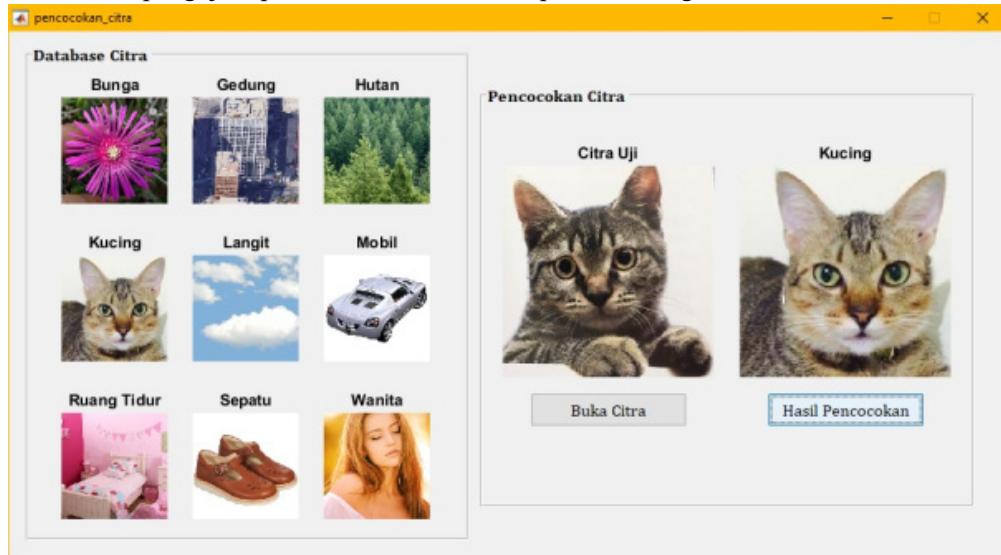
5. Melakukan pengujian pencocokan citra terhadap citra gedung



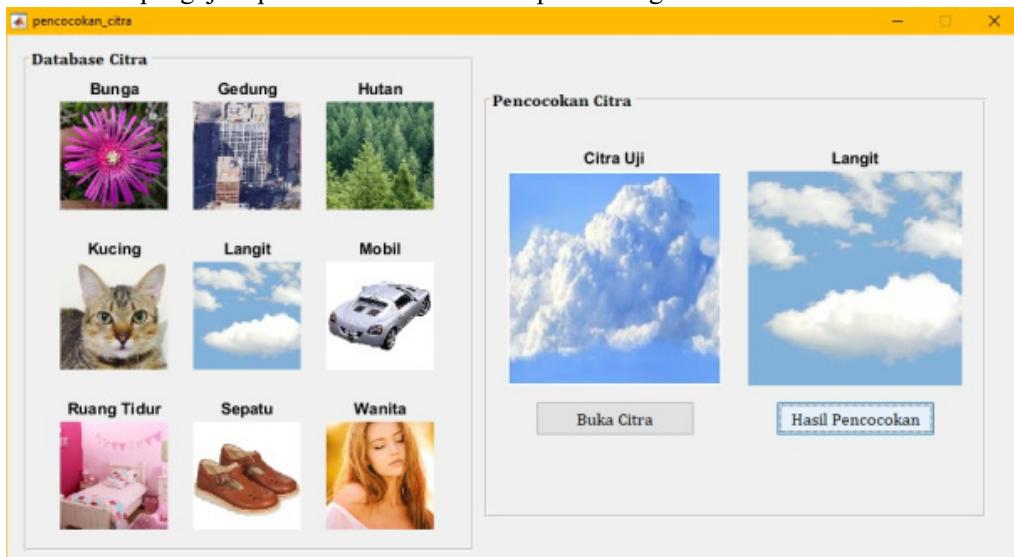
6. Melakukan pengujian pencocokan citra terhadap citra hutan



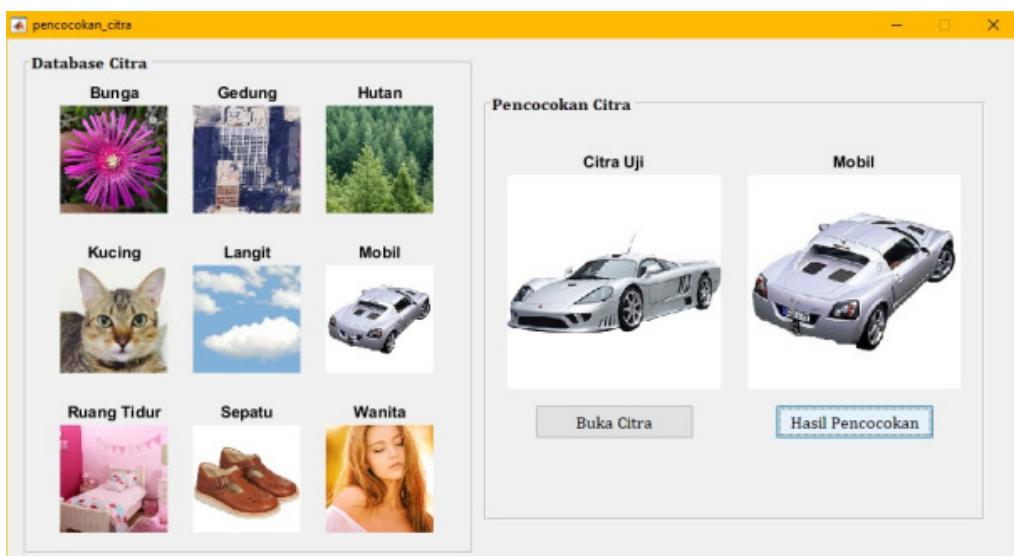
7. Melakukan pengujian pencocokan citra terhadap citra kucing



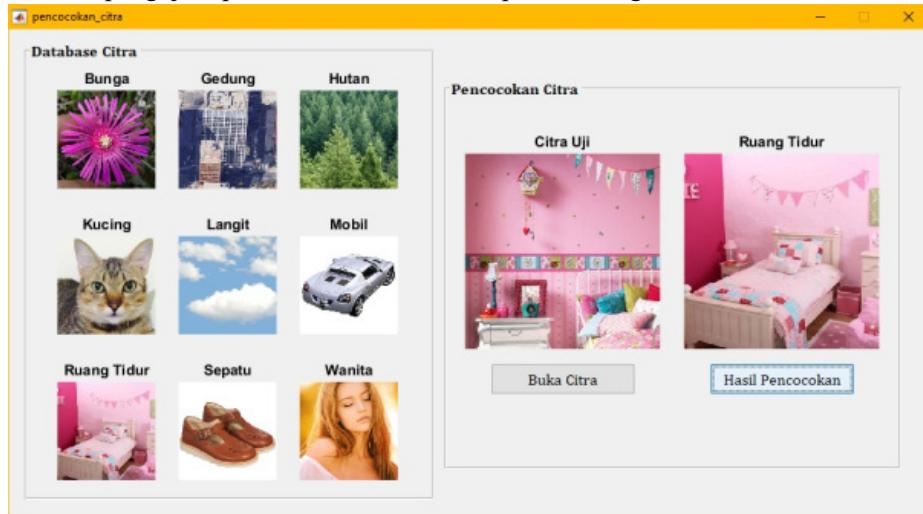
8. Melakukan pengujian pencocokan citra terhadap citra langit



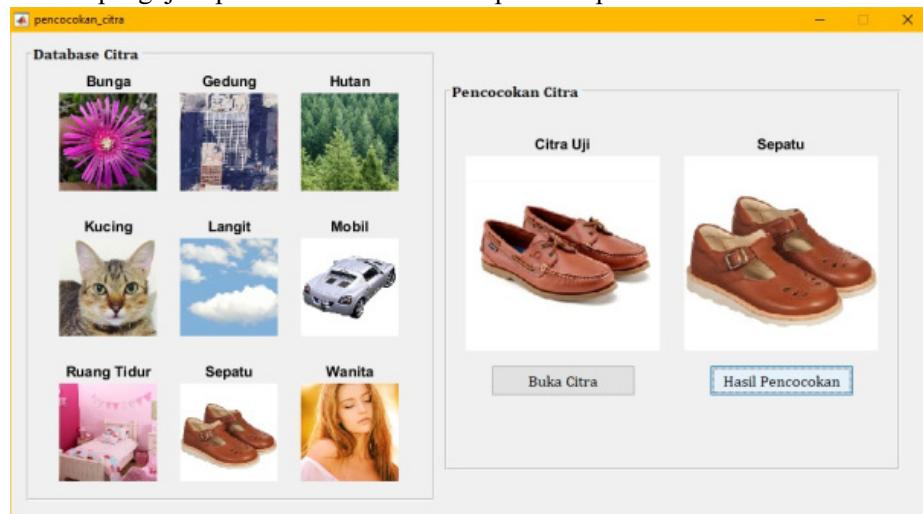
9. Melakukan pengujian pencocokan citra terhadap citra mobil



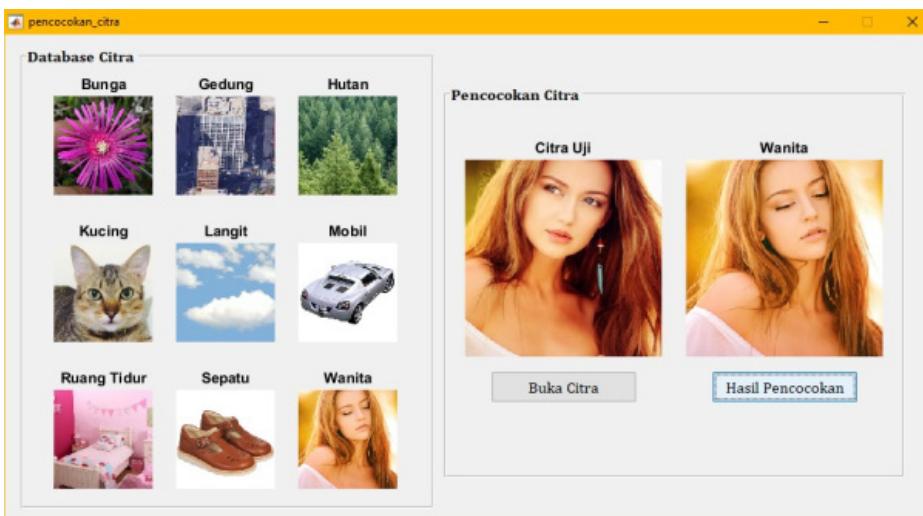
10. Melakukan pengujian pencocokan citra terhadap citra ruang tidur



11. Melakukan pengujian pencocokan citra terhadap citra sepatu



12. Melakukan pengujian pencocokan citra terhadap citra wanita



Dari hasil pengujian terhadap kesembilan buah citra di atas, tampak bahwa setiap citra yang diujikan cocok dengan salah satu citra sejenis yang ada pada database. Hal ini menunjukkan bahwa penghitungan jarak

euclidean berdasarkan ciri warna pada ruang warna *HSV* cukup baik diimplementasikan dalam sistem pencocokan citra.

Tampilan source code matlab pada pemrograman di atas adalah sebagai berikut:

```
1 function varargout = pencocokan_citra(varargin)
2 % PENCOCOKAN_CITRA MATLAB code for pencocokan_citra.fig
3 %     PENCOCOKAN_CITRA, by itself, creates a new PENCOCOKAN_CITRA or raises the
4 % existing
5 %     singleton*.
6 %
7 %     H = PENCOCOKAN_CITRA returns the handle to a new PENCOCOKAN_CITRA or the
8 % handle to
9 %     the existing singleton*.
10 %
11 %     PENCOCOKAN_CITRA('CALLBACK', hObject, eventData, handles,...) calls the local
12 % function named CALLBACK in PENCOCOKAN_CITRA.M with the given input argument
13 %
14 %     PENCOCOKAN_CITRA('Property','Value',...) creates a new PENCOCOKAN_CITRA or
15 % raises the
16 %     existing singleton*. Starting from the left, property value pairs are
17 % applied to the GUI before pencocokan_citra_OpeningFcn gets called. An
18 % unrecognized property name or invalid value makes property application
19 % stop. All inputs are passed to pencocokan_citra_OpeningFcn via varargin.
20 %
21 % *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
22 % instance to run (singleton)".
23 %
24 % See also: GUIDE, GUIDATA, GUIHANDLES
25 %
26 % Edit the above text to modify the response to help pencocokan_citra
27 %
28 % Last Modified by GUIDE v2.5 12-Jul-2017 05:08:25
29 %
30 % Begin initialization code - DO NOT EDIT
31 gui_Singleton = 1;
32 gui_State = struct('gui_Name',          mfilename, ...
33                   'gui_Singleton',    gui_Singleton, ...
34                   'gui_OpeningFcn',   @pencocokan_citra_OpeningFcn, ...
35                   'gui_OutputFcn',    @pencocokan_citra_OutputFcn, ...
36                   'gui_LayoutFcn',    [], ...
37                   'gui_Callback',     []);
38 if nargin && ischar(varargin{1})
39     gui_State.gui_Callback = str2func(varargin{1});
40 end
41 if nargout
42     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
43 else
44     gui_mainfcn(gui_State, varargin{:});
45 end
46 % End initialization code - DO NOT EDIT
47
48
49 % --- Executes just before pencocokan_citra is made visible.
50 function pencocokan_citra_OpeningFcn(hObject, eventdata, handles, varargin)
51 % This function has no output args, see OutputFcn.
52 % hObject    handle to figure
53 % eventdata   reserved - to be defined in a future version of MATLAB
54 % handles    structure with handles and user data (see GUIDATA)
```

```

55 % varargin    command line arguments to pencocokan_citra (see VARARGIN)
56
57 % Choose default command line output for pencocokan_citra
58 handles.output = hObject;
59
60 % Update handles structure
61 movegui(hObject,'center');
62
63 axes(handles.axes1);
64 handles.Img1 = imread('data base\bunga 1.jpg');
65 imshow(handles.Img1);
66 title('Bunga')
67
68 axes(handles.axes2);
69 handles.Img2 = imread('data base\gedung 1.jpg');
70 imshow(handles.Img2);
71 title('Gedung')
72
73 axes(handles.axes3);
74 handles.Img3 = imread('data base\hutan 1.jpg');
75 imshow(handles.Img3);
76 title('Hutan')
77
78 axes(handles.axes4);
79 handles.Img4 = imread('data base\kucing 1.jpg');
80 imshow(handles.Img4);
81 title('Kucing')
82
83 axes(handles.axes5);
84 handles.Img5 = imread('data base\langit 1.jpg');
85 imshow(handles.Img5);
86 title('Langit')
87
88 axes(handles.axes6);
89 handles.Img6 = imread('data base\mobil 1.jpg');
90 imshow(handles.Img6);
91 title('Mobil')
92
93 axes(handles.axes7);
94 handles.Img7 = imread('data base\ruang tidur 1.jpg');
95 imshow(handles.Img7);
96 title('Ruang Tidur')
97
98 axes(handles.axes8);
99 handles.Img8 = imread('data base\sepatu 1.jpg');
100 imshow(handles.Img8);
101 title('Sepatu')
102
103 axes(handles.axes9);
104 handles.Img9 = imread('data base\wanita 1.jpg');
105 imshow(handles.Img9);
106 title('Wanita')
107 guidata(hObject, handles);
108
109
110 % UIWAIT makes pencocokan_citra wait for user response (see UIRESUME)
111 % uiwait(handles.fig)
112
113 function varargout = pencocokan_citra_OutputFcn(hObject, eventdata, handles)
114 % varargout cell array for returning output args (see VARARGOUT);

```

```

115 % hObject    handle to figure
116 % eventdata reserved - to be defined in a future version of MATLAB
117 % handles    structure with handles and user data (see GUIDATA)
118
119 % Get default command line output from handles structure
120 varargout{1} = handles.output;
121
122
123 % --- Executes on button press in pushbutton1.
124 function pushbutton1_Callback(hObject, eventdata, handles)
125 % hObject    handle to pushbutton1 (see GCBO)
126 % eventdata reserved - to be defined in a future version of MATLAB
127 % handles    structure with handles and user data (see GUIDATA)
128 [filename, pathname] = uigetfile('*.*');
129
130 if ~isequal(filename,0)
131     Img = imread(fullfile(pathname,filename));
132     axes(handles.axes10)
133     imshow(Img);
134     title('Citra Uji')
135 else
136     return
137 end
138
139 handles.Img = Img;
140 guidata(hObject, handles);
141
142
143 % --- Executes on button press in pushbutton2.
144 function pushbutton2_Callback(hObject, eventdata, handles)
145 % hObject    handle to pushbutton2 (see GCBO)
146 % eventdata reserved - to be defined in a future version of MATLAB
147 % handles    structure with handles and user data (see GUIDATA)
148 Img = rgb2HSV(handles.Img);
149 Img1 = rgb2HSV(handles.Img1);
150 Img2 = rgb2HSV(handles.Img2);
151 Img3 = rgb2HSV(handles.Img3);
152 Img4 = rgb2HSV(handles.Img4);
153 Img5 = rgb2HSV(handles.Img5);
154 Img6 = rgb2HSV(handles.Img6);
155 Img7 = rgb2HSV(handles.Img7);
156 Img8 = rgb2HSV(handles.Img8);
157 Img9 = rgb2HSV(handles.Img9);
158
159 dist1 = mean(sqrt(sum(sum((Img-Img1).^2)))); % calculate distance between original image and first image
160 dist2 = mean(sqrt(sum(sum((Img-Img2).^2)))); % calculate distance between original image and second image
161 dist3 = mean(sqrt(sum(sum((Img-Img3).^2)))); % calculate distance between original image and third image
162 dist4 = mean(sqrt(sum(sum((Img-Img4).^2)))); % calculate distance between original image and fourth image
163 dist5 = mean(sqrt(sum(sum((Img-Img5).^2)))); % calculate distance between original image and fifth image
164 dist6 = mean(sqrt(sum(sum((Img-Img6).^2)))); % calculate distance between original image and sixth image
165 dist7 = mean(sqrt(sum(sum((Img-Img7).^2)))); % calculate distance between original image and seventh image
166 dist8 = mean(sqrt(sum(sum((Img-Img8).^2)))); % calculate distance between original image and eighth image
167 dist9 = mean(sqrt(sum(sum((Img-Img9).^2)))); % calculate distance between original image and ninth image
168
169 jarak = [dist1;dist2;dist3;dist4;dist5;dist6;dist7;dist8;dist9];
170 [~,n] = min(jarak);
171
172 switch n
173     case 1
174         axes(handles.axes11);
175         handles.Img2 = imread('data base\bunga 1.jpg');

```

```

175         imshow(handles.Img2);
176         title('Bunga')
177     case 2
178         axes(handles.axes11);
179         handles.Img3 = imread('data base\gedung 1.jpg');
180         imshow(handles.Img3);
181         title('Gedung')
182     case 3
183         axes(handles.axes11);
184         handles.Img4 = imread('data base\hutan 1.jpg');
185         imshow(handles.Img4);
186         title('Hutan')
187     case 4
188         axes(handles.axes11);
189         handles.Img1 = imread('data base\kucing 1.jpg');
190         imshow(handles.Img1);
191         title('Kucing')
192     case 5
193         axes(handles.axes11);
194         handles.Img5 = imread('data base\langit 1.jpg');
195         imshow(handles.Img5);
196         title('Langit')
197     case 6
198         axes(handles.axes11);
199         handles.Img6 = imread('data base\mobil 1.jpg');
200         imshow(handles.Img6);
201         title('Mobil')
202     case 7
203         axes(handles.axes11);
204         handles.Img7 = imread('data base\ruang tidur 1.jpg');
205         imshow(handles.Img7);
206         title('Ruang Tidur')
207     case 8
208         axes(handles.axes11);
209         handles.Img8 = imread('data base\sepatu 1.jpg');
210         imshow(handles.Img8);
211         title('Sepatu')
212     case 9
213         axes(handles.axes11);
214         handles.Img9 = imread('data base\wanita 1.jpg');
215         imshow(handles.Img9);
216         title('Wanita')
217 end

```

# TEORI PENGUKURAN JARAK

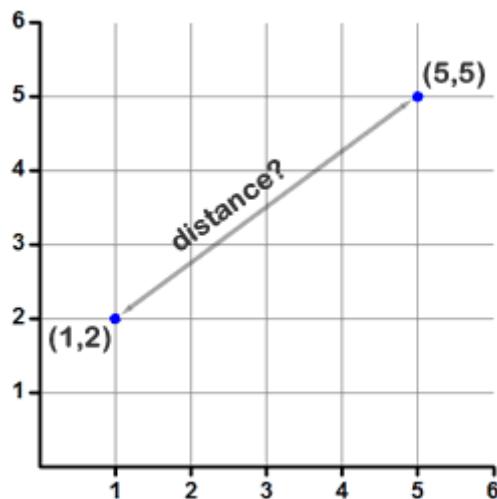
## Teori Euclidean Distance

*Euclidean distance* adalah perhitungan jarak dari 2 buah titik dalam *Euclidean space*. Euclidean space diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini berkaitan dengan Teorema Phytagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tapi juga sederhana jika diterapkan pada dimensi yang lebih tinggi.

### Pada 1 dimensi

Semisal ingin menghitung jarak Euclidean 1 dimensi. Titik pertama adalah 4, titik kedua adalah -10. Caranya adalah kurangkan -10 dengan 4. sehingga menghasilkan -14. Cari nilai absolut dari nilai -14 dengan cara mem pangkatkannya sehingga mendapat nilai 196. Kemudian diakarkan sehingga mendapatkan nilai 14. Sehingga jarak euclidean dari 2 titik tersebut adalah 14.

### Pada 2 dimensi



Caranya hampir sama. Misalkan titik pertama mempunyai kordinat (1,2). Titik kedua ada di kordinat (5,5). Caranya adalah kurangkan setiap kordinat titik kedua dengan titik yang pertama. Yaitu, (5-1, 5-2) sehingga menjadi (4,3). Kemudian pangkatkan masing-masing sehingga memperoleh (16,9). Kemudian tambahkan semuanya sehingga memperoleh nilai  $16+9 = 25$ . Hasil ini kemudian diakarkan menjadi 5. Sehingga jarak euclideanya adalah 5.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Rumus Euclid

Sehingga dari Formula diatas kita dapat implementasi menjadi :

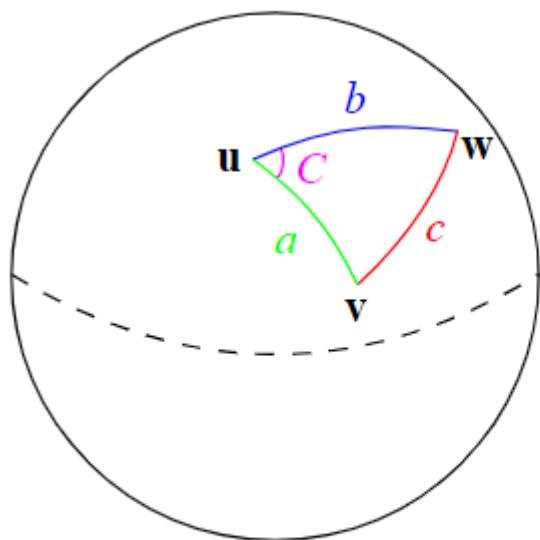
$$\text{Jarak} = \sqrt{(Lat_1 - Lat_2)^2 + (Long_1 - Long_2)^2}$$

#### Rumus Jarak Euclidean

Hasil perhitungan (Jarak) diatas masih dalam satuan *decimal degree* (sesuai dengan format longlat yang dipakai) sehingga untuk menyesuaikannya perlu dikalikan dengan **111.319 km** (1 derajat bumi = 111.319 km).

#### Teori Haversine Formula

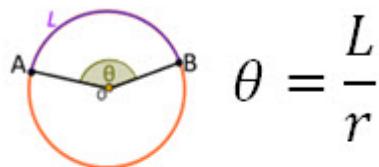
Teorema Haversine Formula adalah sebuah persamaan yang penting dalam bidang navigasi, untuk mencari jarak busur antara dua titik pada bola dari longitude dan latitude. Ini merupakan bentuk persamaan khusus dari trigonometri bola, law of haversines, mencari hubungan sisi dan sudut pada segitiga dalam bidang bola.



Ilustrasi Spherical law of cosines

$$\cos(c) = \cos(a) \cos(b) + \sin(a) \sin(b) \cos(C).$$

Dimana a,b,c ialah jarak yang bersatuan radian/sudut karena berada dalam bidang bola, yang bisa kita korelasikan dengan persamaan busur dibawah ini :



Rumus Busur

Kemudian kita implementasikan persamaan harvesin dibawah ini :

$$\text{haversin}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos(\theta)}{2}$$

Harvesin Formula

Sehingga dari Formula diatas kita dapat implementasi menjadi :

$$\text{Jarak} = 2r \cdot \arcsin \left\{ \sqrt{\sin^2\left(\frac{Lat_1 - Lat_2}{2}\right) + \cos(Lat_1) \cdot \cos(Lat_2) \cdot \sin^2\left(\frac{Long_1 - Long_2}{2}\right)} \right\}$$

Rumus Jarak Harvesin

### *Penggunaan Teorema Euclid dan Teorema Harvesine*

Pertama kita siapkan bahan uji sebagai berikut :

---

## 1. Titik Pertama : Gedung Sate (Bandung)

Long : 107.618633

Lat : -6.901361

## 2. Titik Kedua: Mesjid Raya Lembang (KBB)

Long : 107.618279

Lat : -6.811771

Jarak aktual yang diperoleh dari Gmap ialah 10.05 Km.



Jarak Lurus Bandung Lembang

Baiklah, bahan sudah ada tinggal kita implementasi dengan rumus yang sudah ada. Saya akan sertakan rumus excelnya supaya lebih mudah membuktikannya.

Dengan Teorema Euclid :

$$Jarak = \sqrt{(Lat_1 - Lat_2)^2 + (Long_1 - Long_2)^2}$$

Rumus Jarak Euclidean

$$=((\text{SQRT}((B6-B7)^2+(C6-C7)^2)*111.319))$$

Hasilnya diperoleh Jarak = 9.97 Km

Dengan Teorema Harvesine:

$$Jarak = 2r \cdot \arcsin \left\{ \sqrt{\sin^2 \left( \frac{Lat_1 - Lat_2}{2} \right) + \cos(Lat_1) \cdot \cos(Lat_2) \cdot \sin^2 \left( \frac{Long_1 - Long_2}{2} \right)} \right\}$$

Rumus Jarak Harvesine

$$=(6371.1*((2*\text{ASIN}(\text{SQRT}((\text{SIN}((\text{RADIANS}(B7) - \text{RADIANS}(B6))/2)^2)+\text{COS}(\text{RADIANS}(B7))*\text{COS}(\text{RADIANS}(B6))*(\text{SIN}((\text{RADIANS}(C7) - \text{RADIANS}(C6))/2)^2))))))$$

Hasilnya diperoleh Jarak = 9.96 Km

Hasil perhitungan Excel :

<b>Perhitungan Jarak melalui koordinat</b>			
	Latitude	Longitude	
Gedung Sate	-6.901361	107.618633	
Mesjid Lembang	-6.811771	107.618279	
<b>Euclidean =</b>	9.97	<b>Km</b>	=((\text{SQRT}((C5-C6)^2+(D5-D6)^2)*111.319))
<b>Harvesine =</b>	9.96	<b>Km</b>	=6371.1*((2*\text{ASIN}(\text{SQRT}((\text{SIN}((\text{RADIANS}(B7) - \text{RADIANS}(B6))/2)^2)+\text{COS}(\text{RADIANS}(B7))*\text{COS}(\text{RADIANS}(B6))*(\text{SIN}((\text{RADIANS}(C7) - \text{RADIANS}(C6))/2)^2))))))

Hasil Excel

Dari kedua teorema diatas hasil yang diperoleh tidak berbeda jauh meskipun ada selisih 0.09 dengan hasil Gmap. sehingga cukup ampuh untuk menghitung jarak2 pendek yang kedepannya bisa diakumulasi untuk memperoleh jarak sesuai jalur yang dilalui.

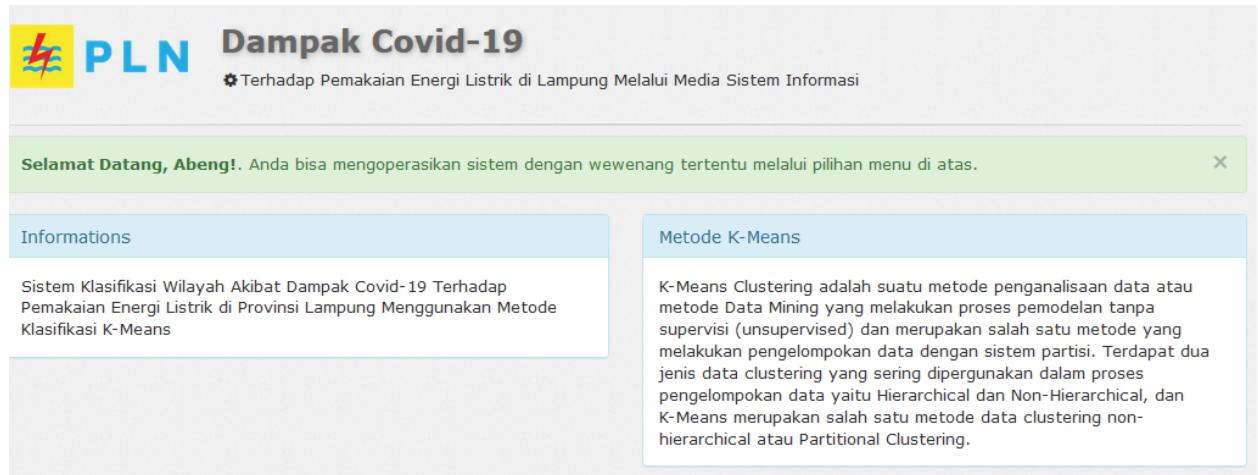
Reff :

<https://blogs.itb.ac.id/anugraha/2014/09/10/teori-pengukuran-jarak/>

**Nama** : Bhijanta Wyasa WM  
**NIM** : 202420019  
**Kelar** : MTI 23  
**Dosen** : Tri Basuki Kurniawan , S.Kom., M.Eng. Ph.D  
**Source** : <https://clusterisasi.com/k-means/admin/maps>

1. Silahkan cari satu tutorial yang membahas tentang pengukuran jarak antar data. Buat ringkasan dari tutorial tersebut dan tulis dalam format ms word, lalu kumpulkan sebelum batas waktu. Sertakan sumber link tutorial yang diringkas.

Pada kesempatan kali ini saya ingin mencoba tentang pengukuran jarak antar data pada sistem klusterisasi data pemakaian listrik sebelum dan sesudah Covid 19 di Area Prov Lampung,

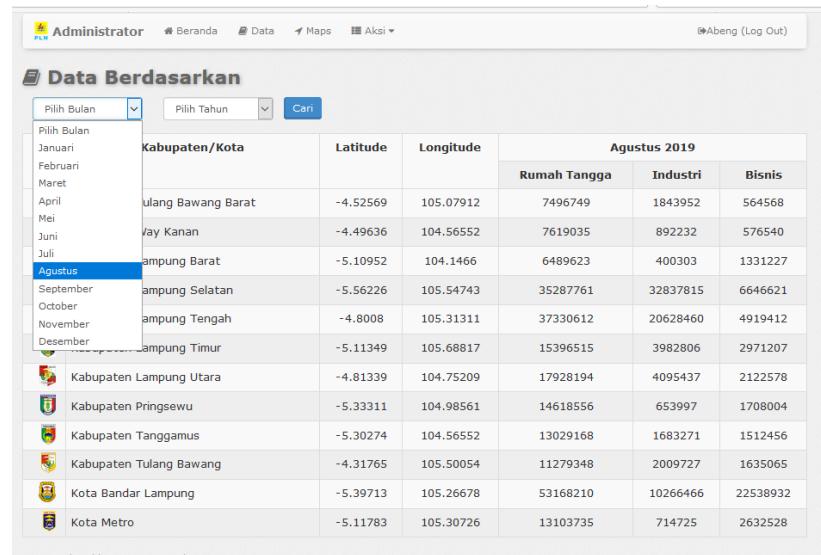


The screenshot shows a web page with the PLN logo and title 'Dampak Covid-19'. A message says 'Selamat Datang, Abeng! Anda bisa mengoperasikan sistem dengan wewenang tertentu melalui pilihan menu di atas.' Below are two sections: 'Informations' and 'Metode K-Means'.

**Informations:**  
Sistem Klasifikasi Wilayah Akibat Dampak Covid-19 Terhadap Pemakaian Energi Listrik di Provinsi Lampung Menggunakan Metode Klasifikasi K-Means

**Metode K-Means:**  
K-Means Clustering adalah suatu metode penganalisaan data atau metode Data Mining yang melakukan proses pemodelan tanpa supervisi (unsupervised) dan merupakan salah satu metode yang melakukan pengelompokan data dengan sistem partisi. Terdapat dua jenis data clustering yang sering dipergunakan dalam proses pengelompokan data yaitu Hierarchical dan Non-Hierarchical, dan K-Means merupakan salah satu metode data clustering non-hierarchical atau Partitional Clustering.

Pada sistem ini didapatkan data pemakaian komsumsi energi listrik dari sebelum Covid 19 ( Agustus – Desember 2019 ) sampai dengan setelah covid 19 ( Jan – Mei 2020 ).



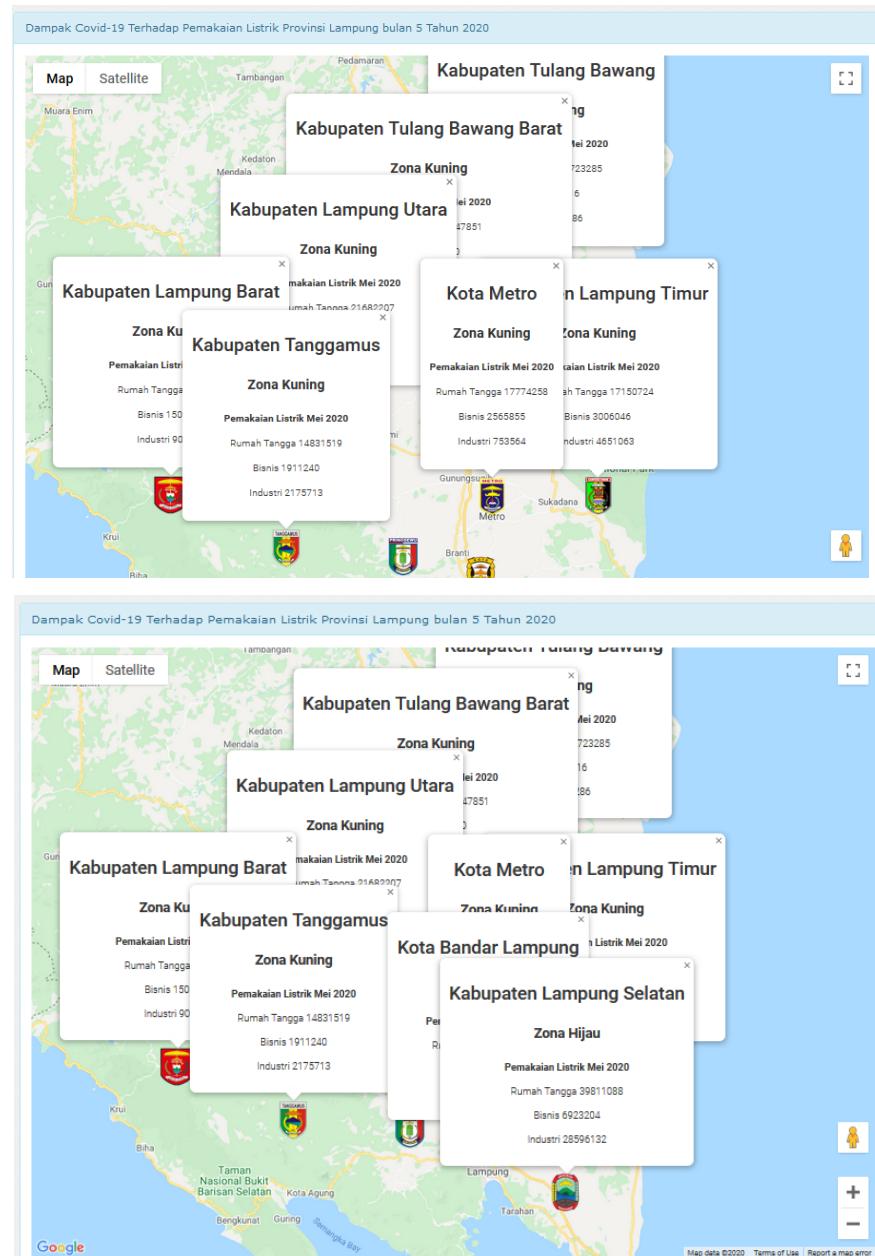
The screenshot shows a table titled 'Data Berdasarkan' with columns for Kabupaten/Kota, Latitude, Longitude, and three categories for August 2019: Rumah Tangga, Industri, and Bisnis. The table includes icons for each district.

Pilih Bulan	Pilih Tahun	Cari	Kabupaten/Kota	Latitude	Longitude	Agustus 2019		
						Rumah Tangga	Industri	Bisnis
Januari			Lampung Barat	-4.52569	105.07912	7496749	1843952	564568
Februari			Lampung Selatan	-4.49636	104.56552	7619035	892232	576540
Maret			Lampung Tengah	-4.8008	105.31311	37330612	20628460	4919412
April			Lampung Timur	-5.11349	105.68817	15396515	3982806	2971207
Mei			Kabupaten Lampung Utara	-4.81339	104.75209	17928194	4095437	2122578
Juni			Kabupaten Pringsewu	-5.33311	104.98561	14618556	653997	1708004
Juli			Kabupaten Tanggamus	-5.30274	104.56552	13029168	1683271	1512456
Agustus			Kabupaten Tulang Bawang	-4.31765	105.50054	11279348	2009727	1635065
September			Kota Bandar Lampung	-5.39713	105.26678	53168210	10266466	22538932
Oktober			Kota Metro	-5.11783	105.30726	13103735	714725	2632528
November								
Desember								

Page rendered in 0.0122 seconds

**Nama** : Bhijanta Wyasa WM  
**NIM** : 202420019  
**Kelar** : MTI 23  
**Dosen** : Tri Basuki Kurniawan , S.Kom., M.Eng. Ph.D  
**Source** : <https://clusterisasi.com/k-means/admin/maps>

Dari keseluruhan data yang sudah masuk dalam sistem dibuat klasifikasi grade pemakaian komsumsi listrik dimasing – masing kab kota yang ada di Prov Lampung. Pada hal ini lakukan clusterisasi sesuai dengan klasifikasi pemakaian ( Hijau, Kuning, Merah ).

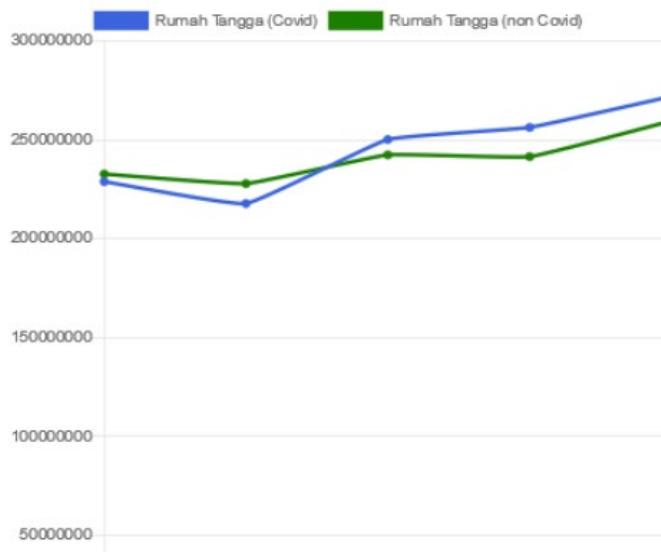


Kemudian setelah dilakukan klasifikasi terkait dengan pemakaian tanpa melihat setelah covid 19 atau belum sekarang akan dilakukan pengukuran data dari data pemakaian energi listrik sebelum dan sesudah adanya covid 19 di Prov Lampung. Adapun untuk klasifikasinya dibagi menjadi :

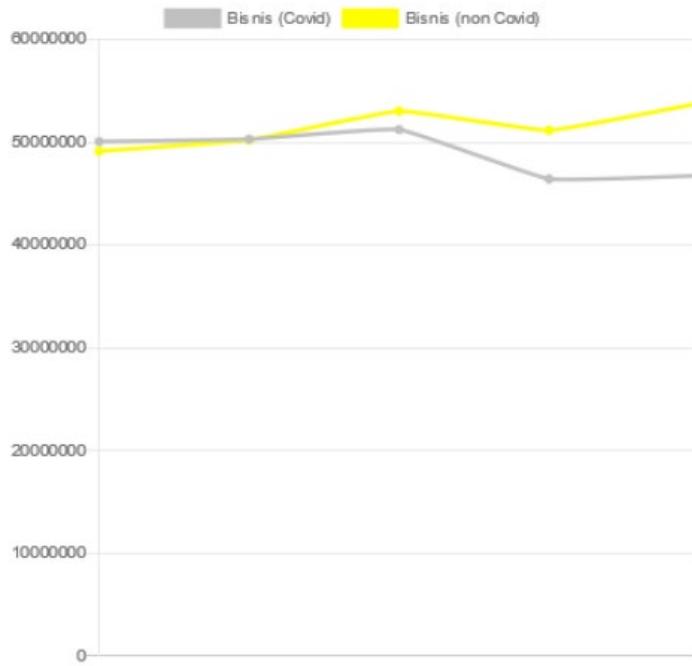
Nama : Bhijanta Wyasa WM  
NIM : 202420019  
Kelas : MTI 23  
Dosen : Tri Basuki Kurniawan , S.Kom., M.Eng. Ph.D  
Source : <https://clusterisasi.com/k-means/admin/maps>

1. Rumah Tangga
2. Bisnis
3. Industri

#### Pengukuran Jarak Data pada segment Rumah Tangga :

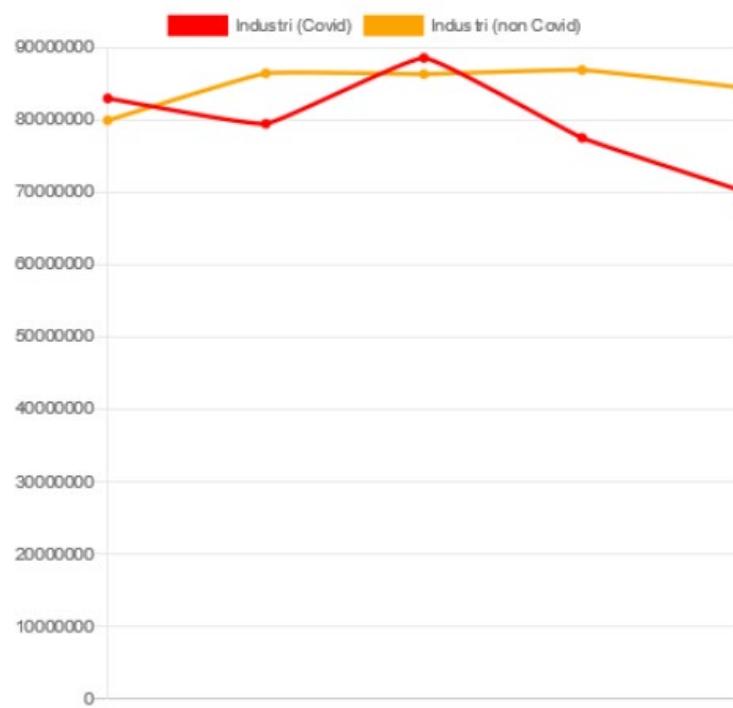


#### Pengukuran Jarak Data pada segment Bisnis :



Nama : Bhijanta Wyasa WM  
NIM : 202420019  
Kelas : MTI 23  
Dosen : Tri Basuki Kurniawan , S.Kom., M.Eng. Ph.D  
Source : <https://clusterisasi.com/k-means/admin/maps>

Pengukuran Jarak Data pada segment Industri :



Sehingga pengukuran jarak antar data pada kasus ini diperoleh dari data pemakaian sumber energi listrik yang ada di Prov Lampung. Untuk implementasi dari data mining menggunakan algoritma K-Means dalam melakukan Pengukuran Jarak Antar Data.

Ada beberapa kelebihan pada algoritma k-means, yaitu :

- Mudah untuk diimplementasikan dan dijalankan.
- Waktu yang dibutuhkan untuk menjalankan pembelajaran ini relatif cepat.
- Mudah untuk diadaptasi.
- Umum digunakan.

Algoritma k-means memiliki beberapa kelebihan, namun ada kekurangannya juga. Kekurangan dari algoritma tersebut yaitu :

1. Sebelum algoritma dijalankan, k buah titik diinisialisasi secara random sehingga pengelompokan data yang dihasilkan dapat berbeda-beda . Jika nilai random untuk inisialisasi kurang baik, maka pengelompokan yang dihasilkan pun menjadi kurang optimal.

**Nama** : Bhijanta Wyasa WM  
**NIM** : 202420019  
**Kelar** : MTI 23  
**Dosen** : Tri Basuki Kurniawan , S.Kom., M.Eng. Ph.D  
**Source** : <https://clusterisasi.com/k-means/admin/maps>

2. Dapat terjebak dalam masalah yang disebut curse of dimensionality. Hal ini dapat terjadi jika data pelatihan memiliki dimensi yang sangat tinggi (Contoh jika data pelatihan terdiri dari 2 atribut maka dimensinya adalah 2 dimensi. Namun jika ada 20 atribut, maka akan ada 20 dimensi). Salah satu cara kerja algoritma ini adalah mencari jarak terdekat antara k buah titik dengan titik lainnya. Jika mencari jarak antar titik pada 2 dimensi, masih mudah dilakukan. Namun bagaimana mencari jarak antar titik jika terdapat 20 dimensi. Hal ini akan menjadi sulit.
3. Jika hanya terdapat beberapa titik sampel data, maka cukup mudah untuk menghitung dan mencari titik terdekat dengan k titik yang diinisialisasi secara random. Namun jika terdapat banyak sekali titik data (misalnya satu miliar buah data), maka perhitungan dan pencarian titik terdekat akan membutuhkan waktu yang lama. Proses tersebut dapat dipercepat, namun dibutuhkan struktur data yang lebih rumit seperti kD-Tree atau hashing.

Nama : Cornelia Tri Wahyuni  
NIM : 202420044  
Mata Kuliah : Advanced Database

Tahap-tahap pengukuran jarak antar data dengan algoritma K-Means clustering adalah sebagai berikut :

1. Tentukan jumlah cluster lalu menginput data yang digunakan untuk menentukan nilai rata-rata data dalam satu cluster dan menentukan jarak dari setiap data ke centroid.
2. Alokasikan data ke dalam cluster secara random atau acak
3. Hitung centroid/rata-rata dari data yang ada di masing-masing cluster, nilai centroid pada k-means digunakan sebagai pusat cluster. Dengan menentukan anggota cluster secara acak pada tahap sebelumnya, maka terbentuk iterasi awal sebagai pusat cluster acak.
4. Alokasikan masing-masing data ke centroid/rata-rata terdekat, pada tahap ini hasil centroid dari setiap cluster sudah diketahui, kemudian data dialokasikan pada centroid terdekat berdasarkan nilai jarak similarity data terhadap centroid. Jarak similarity dari data ke centroid pada masing-masing cluster diperoleh dari perhitungan Euclidean Distance. Kemudian nilai jarak setiap data ke centroid cluster dibandingkan, dan data point menjadi anggota dari cluster berdasarkan jarak data ke centroid terdekat.
5. Konvergen, mengalokasikan data point ke centroid dengan nilai jarak terdekat, dengan menguji apakah cluster yang terbentuk telah membentuk cluster yang konvergen atau tidak. Cluster dinyatakan konvergen jika anggota dari masing-masing cluster yang terbentuk tidak mengalami perubahan anggota. Tetapi jika masih terjadi perubahan anggota cluster, maka akan kembali dilakukan tahapan menghitung centroid dari masing-masing cluster yang terbentuk dan diikuti dengan perhitungan nilai similarity ke centroid yang baru terbentuk. Proses tersebut terus berulang sampai hasil cluster konvergen.

Sumber : <https://informatikalogi.com/algoritma-k-means-clustering/>  
<https://ejurnal.gunadarma.ac.id/index.php/tekno/article/view/1599/1358>  
[https://www.youtube.com/watch?v=8mXL2z3lg\\_o&t=11s](https://www.youtube.com/watch?v=8mXL2z3lg_o&t=11s)

Nama : Cynthia Anisa Agatha

NIM : 202420022

Membuat sebuah tutorial berupa resume mengenai pengukuran jarak antar data

Sumber mengutip dari

- [mulaab.github.io/datamining/memahami-data/#mengukur-jarak-data](https://mulaab.github.io/datamining/memahami-data/#mengukur-jarak-data)
- Ning, Pang. 2006. *Introduction to Data Mining*. Boston.

## **Mengukur Jarak Data**

Pengukuran jarak antar data memiliki peranan penting, yang dapat membantu dalam bidang *data mining* diantaranya yaitu algoritma *clustering* dengan berbasis jarak. Misalkan seperti *K-Mean*, *K-medoidm*, *Fuzzy c-mean* dan *rough clustering* dimana, proses ini berdasarkan dari jarak untuk mengelompokkan.

Terdapat beberapa pengukuran jarak antar data, yaitu:

- 1) Mengukur jarak dengan tipe atribut nominal
- 2) Mengukur jarak dengan tipe atribut ordinal
- 3) Mengukur jarak dengan tipe atribut interval
- 4) Mengukur jarak dengan tipe atribut rasio (campuran)

### **Pengukuran jarak tipe numerik**

Nilai dari atribut nominal itu sendiri memiliki nama yang berbeda, dimana nilai nominal menyediakan informasi terbatas dalam membedakan satu objek dengan objek lainnya. Operasi matematis yang dilaksanakan pada proses ini yaitu mode, entropy, contingency, korelasi dan tes xkuadrat. Contoh variabel yang sesuai dengan tipe nominal ini diantaranya adalah kode pos, nomor induk pegawai, warna mata dan jenis kelamin. Terdapat berbagai macam metode pengukuran jarak dalam kategori tipe numerik. Beberapa diantaranya yaitu:

1. *Minkowski Distance*

Kelompok *Minkowski* diantaranya adalah *Euclidean Distance* dan *Manhattan Distance*, yaitu sebuah kasus yang khusus dari *Minkowski Distance*. *Minkowski* dinyatakan dengan:

$$d_{min} = (\sum_{i=1}^n |x_i - y_i|^m)^{\frac{1}{m}}, m \geq 1$$

Dimana  $m$  adalah bilangan riil positif dan  $x_i$  dan  $y_i$  merupakan dua vektor yang berada dalam dimensi  $n$ . Sebelum melakukan implementasi ukuran jarak *Minkowski* pada model *clustering* data atribut perlu dilakukan normalisasi terlebih dahulu untuk menghindari terlalu banyaknya atribut dengan skala data cukup besar.

## 2. Manhattan Distance

*Manhattan Distance* merupakan kasus khusus dari jarak *Minkowski distance* pada  $m = 1$ . Serupa dengan *Minkowski Distance*, *Manhattan Distance* lebih rentan pada *outlier*. Apabila pengukuran ini digunakan dalam Algoritma *Clustering*, maka bentuk *cluster* menjadi *hyper-reactangular*. Pengukuran didefinisikan sebagai berikut:

$$d_{man} = \sum_{i=1}^n |x_i - y_i|$$

## 3. Euclidean Distance

Dsitanse yang sering digunakan untuk data numerik yaitu *Euclidean Distance*. *Eucledian Distance* bekerja dapat bekerja dengan baik untuk kumpulan data yang solid. Kendatipun, *Euclidean Distance* sering digunakan, ia pun juga memiliki kelemahan yaitu:

- a) Apabila dua vektor data tidak mempunyai nilai atribut yang sama, besar kemungkinannya jarak yang dihasilkan lebih kecil dibandingkan pasangan vektor data lainnya yang terkandung nilai atribut yang sama.

## 4. Average Distance

Terlepas dari kelemahan yang dimiliki *Eucledian Distance*, Average Distance merupakan sebuah tipe transformasi dari *Euclidian Distance* untuk mengubah hasil. Untuk dua titik  $x, y$  dalam dimensi  $n$ , *Average Distance* dapat didefinisikan sebagai berikut:

$$d_{ave} = (\frac{1}{n} \sum_{i=1}^n (x_i - y_i)^2)^{\frac{1}{2}}$$

## 5. Weighted Euclidean Distance

Berdasarkan tingkat kepentingan per atribut yang ditentukan, maka *Weighted Euclidean Distance* merupakan transformasi lainnya dari *Euclidean Distance* yang juga dapat digunakan. Pengukuran dapat dirumuskan dengan

$$d_{we} = \left( \sum_{i=1}^n w_i (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

Dimana  $w_i$  merupakan bobot yang diberikan terhadap atribut ke  $i$ .

#### 6. Chord Distance

*Chord Distance* adalah salah satu metode pengukuran jarak transformasi *Euclidean Distance* untuk mengurangi kekurangan dari *Euclidean Distance*. Metode pengukuran jarak ini dengan menghitung dari data yang belum ternormalisasi. Chord distance didefinisikan dengan

$$d_{chord} = (2 - 2 \frac{\sum_{i=1}^n x_i y_i}{\|x\|^2 \|y\|^2})^{\frac{1}{2}}$$

Dimana  $\|x\|^2$  adalah  $L^2 - norm\|x\|^2 = \sqrt{\sum_{i=1}^n x_i^2}$

#### Pengukuran jarak dengan tipe ordinal

Nilai dari tipe atribut ordinal menyediakan informasi yang cukup dalam memberikan instruksi sebuah objek ( $>$  dan  $<$ ). Operasi matematis yang dapat dipergunakan pada tipe ordinal diantaranya median, percentil, korelasi peringkat dan lain-lain. Contohnya berupa nomor jalan dan nilai. Metode yang terdapat dalam kategori tipe ordinal salah satunya yaitu *Jaccard Coefficient*.

#### Pengukuran jarak dengan tipe interval

Untuk atribut interval yaitu perbedaan diantara nilai yang memiliki arti secara khusus misalkan bagian pengukuran yang ada (+, -). Operasi matematis yang dapat dipergunakan pada tipe interval ini yaitu mean, standar deviasi, *Pearson*, korelasi dan tes *t* dan *F*. Contoh dari variabel yang ada dalam kehidupan kita sehari-hari seperti tanggal pada kalender, suhu dalam *Celcius* dan *Fahrenheit*.

## **Pengukuran jarak dengan tipe rasio**

Untuk variabel rasio, kedua perbedaan dan rasio memiliki peran amat penting (\*, /). Operasi matematis pada pengukuran jarak dengan tipe rasio yang dapat dipergunakan *geometric mean, harmonic mean, percent variation*. Untuk contoh variabel yang sering kita jumpai pada umumnya misalnya umur, durasi, massa dan masih banyak lagi.

Nama : Efrik Kartono Ahsa  
NIM : 202420030  
MK : Advanced Database  
LINK : <https://blogs.itb.ac.id/anugraha/2014/09/10/teori-pengukuran-jarak/>

## Pengukuran Jarak Antar Data

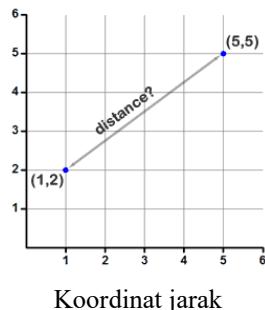
### A. Teori Euclidean Distance

*Euclidean distance* adalah perhitungan jarak dari 2 buah titik dalam *Euclidean space*. Euclidean space diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini berkaitan dengan Teorema Phytagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tapi juga sederhana jika diterapkan pada dimensi yang lebih tinggi.

#### Pada 1 dimensi

Semisal ingin menghitung jarak Euclidean 1 dimensi. Titik pertama adalah 4, titik kedua adalah -10. Caranya adalah kurangkan -10 dengan 4. sehingga menghasilkan -14. Cari nilai absolut dari nilai -14 dengan cara mem pangkatkannya sehingga mendapat nilai 196. Kemudian diakarkan sehingga mendapatkan nilai 14. Sehingga jarak euclidean dari 2 titik tersebut adalah 14.

#### Pada 2 dimensi



Caranya hampir sama. Misalkan titik pertama mempunyai kordinat (1,2). Titik kedua ada di kordinat (5,5). Caranya adalah kurangkan setiap kordinat titik kedua dengan titik yang pertama. Yaitu, (5-1,5-2) sehingga menjadi (4,3). Kemudian pangkatkan masing-masing sehingga memperoleh (16,9). Kemudian tambahkan semuanya sehingga memperoleh nilai  $16+9 = 25$ . Hasil ini kemudian diakarkan menjadi 5. Sehingga jarak euclideanya adalah 5.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Rumus Euclid

Sehingga dari Formula diatas kita dapat implementasi menjadi :

$$\text{Jarak} = \sqrt{(\text{Lat}_1 - \text{Lat}_2)^2 + (\text{Long}_1 - \text{Long}_2)^2}$$

Rumus Jarak Euclidean

Nama : Efrik Kartono Ahsa

NIM : 202420030

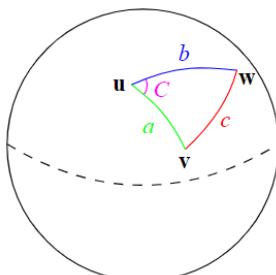
MK : Advanced Database

LINK : <https://blogs.itb.ac.id/anugraha/2014/09/10/teori-pengukuran-jarak/>

Hasil perhitungan (*Jarak*) diatas masih dalam satuan *decimal degree* (sesuai dengan format longlat yang dipakai) sehingga untuk menyesuaikannya perlu dikalikan dengan **111.319 km** (1 derajat bumi = 111.319 km).

## B. Teori Haversine Formula

Teorema Haversine Formula adalah sebuah persamaan yang penting dalam bidang navigasi, untuk mencari jarak busur antara dua titik pada bola dari longitude dan latitude. Ini merupakan bentuk persamaan khusus dari trigonometri bola, law of haversines, mencari hubungan sisi dan sudut pada segitiga dalam bidang bola.

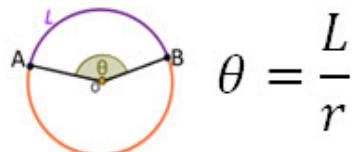


Ilustrasi Spherical law of

$$\cos(c) = \cos(a) \cos(b) + \sin(a) \sin(b) \cos(C).$$

cosines  
Spherical law of cosines

Dimana a,b,c ialah jarak yang bersatuan radian/sudut karena berada dalam bidang bola, yang bisa kita korelasikan dengan persamaan busur dibawah ini :



Rumus Busur

Kemudian kita implementasikan persamaan harvesin dibawah ini :

$$\text{haversin}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos(\theta)}{2}$$

Harvesin Formula

Sehingga dari Formula diatas kita dapat implementasi menjadi :

Nama : Efrik Kartono Ahsa

NIM : 202420030

MK : Advanced Database

LINK : <https://blogs.itb.ac.id/anugraha/2014/09/10/teori-pengukuran-jarak/>

$$Jarak = 2r \cdot \arcsin \left\{ \sqrt{\sin^2 \left( \frac{Lat_1 - Lat_2}{2} \right) + \cos(Lat_1) \cdot \cos(Lat_2) \cdot \sin^2 \left( \frac{Long_1 - Long_2}{2} \right)} \right\}$$

Rumus Jarak Harvesin

## Penggunaan Teorema Euclid dan Teorema Harvesine

Pertama kita siapkan bahan uji sebagai berikut :

### 1. Titik Pertama : *Gedung Sate (Bandung)*

Long : 107.618633

Lat : -6.901361

### 2. Titik Kedua: *Mesjid Raya Lembang (KBB)*

Long : 107.618279

Lat : -6.811771

Jarak aktual yang diperoleh dari Gmap ialah 10.05 Km.



Jarak Lurus Bandung Lembang

Baiklah, bahan sudah ada tinggal kita implementasi dengan rumus yang sudah ada. Saya akan sertakan rumus excelnya supaya lebih mudah membuktikannya.

Dengan Teorema Euclid :

Nama : Efrik Kartono Ahsa  
 NIM : 202420030  
 MK : Advanced Database  
 LINK : <https://blogs.itb.ac.id/anugraha/2014/09/10/teori-pengukuran-jarak/>

$$Jarak = \sqrt{(Lat_1 - Lat_2)^2 + (Long_1 - Long_2)^2}$$

Rumus Jarak Euclide

$$=((\text{SQRT}((B6-B7)^2+(C6-C7)^2)*111.319))$$

Hasilnya diperoleh Jarak = 9.97 Km

Dengan Teorema Harvesine:

$$Jarak = 2r \cdot \arcsin \left\{ \sqrt{\sin^2 \left( \frac{Lat_1 - Lat_2}{2} \right) + \cos(Lat_1) \cdot \cos(Lat_2) \cdot \sin^2 \left( \frac{Long_1 - Long_2}{2} \right)} \right\}$$

Rumus Jarak Harvesin

$$=(6371.1*((2*\text{ASIN}(\text{SQRT}((\text{SIN}((\text{RADIANS}(B7)-\text{RADIANS}(B6))/2)^2)+\text{COS}(\text{RADIANS}(B7))*\text{COS}(\text{RADIANS}(B6))*(\text{SIN}((\text{RADIANS}(C7)-\text{RADIANS}(C6))/2)^2))))))$$

Hasilnya diperoleh Jarak = 9.96 Km

Hasil perhitungan Excel :

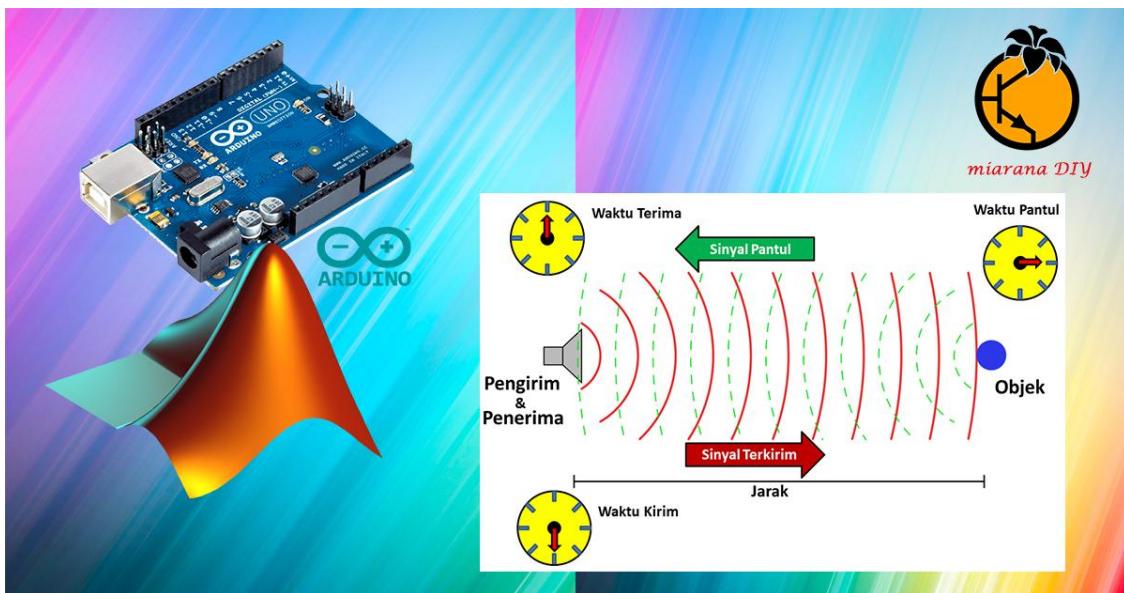
<b>Perhitungan Jarak melalui koordinat</b>			
	Latitude	Longitude	
Gedung Sate	-6.901361	107.618633	
Mesjid Lembang	-6.811771	107.618279	
<b>Euclidean =</b>	9.97	Km	=((\text{SQRT}((C5-C6)^2+(D5-D6)^2)*111.319))
<b>Harvesine =</b>	9.96	Km	= (6371.1*((2*\text{ASIN}(\text{SQRT}((\text{SIN}((\text{RADIANS}(B7)-\text{RADIANS}(B6))/2)^2)+\text{COS}(\text{RADIANS}(B7))*\text{COS}(\text{RADIANS}(B6))*(\text{SIN}((\text{RADIANS}(C7)-\text{RADIANS}(C6))/2)^2))))))

Hasil Excel

Dari kedua teorema diatas hasil yang diperoleh tidak berbeda jauh meskipun ada selisih 0.09 dengan hasil Gmap. sehingga cukup ampuh untuk menghitung jarak2 pendek yang kedepannya bisa diakumulasi untuk memperoleh jarak sesuai jalur yang dilalui.

## TUGAS 01 ADVANCED DATABASE

[TUTORIAL] Ploting Data Real-Time dari Sensor Ultrasonik Menggunakan MATLAB

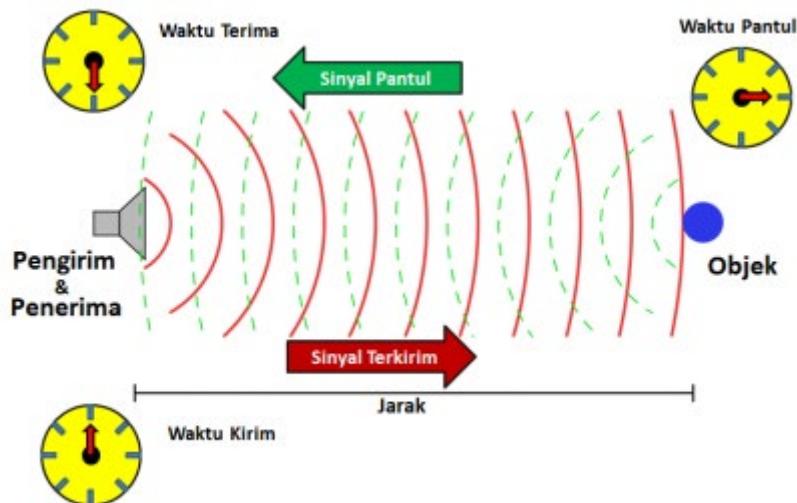


Sensor ultrasonik adalah salah satu sensor yang paling populer digunakan untuk mengukur jarak atau mendeteksi suatu halangan yang berada di depan sensor. Sensor jenis ini menjadi salah satu sensor yang paling banyak digunakan pada event-event perlombaan, seperti Kontes Robot Pemadam Api (KRPAI) yang diselenggarakan setiap tahunnya oleh Direktorat Pendidikan Tinggi (Diktif).

Pada Tulisan kali ini, kita akan membuat suatu antarmuka untuk mendeteksi jarak yang dihasilkan oleh sensor ultrasonik menggunakan MATLAB, dan hasilnya akan ditampilkan berupa grafik batang (bar chart).

### Prinsip Kerja Sensor Ultrasonik

Sensor ultrasonik akan mengirimkan sinyal suara berfrekuensi tinggi berbentuk pulsa dengan interval tertentu. Sinyal ini akan merambat di udara dengan kecepatan suara. Jika sinyal menabrak suatu objek atau halangan, maka sinyal ini akan terpantulkan balik ke sensor. Jarak akan dihitung berdasarkan waktu yang diperlukan untuk mengirim dan menerima sinyal ke sensor. Amatilah **Gambar 1**.



**Gambar 1. Prinsip Kerja Sensor Ultrasonik**

Seperti yang telah kita ketahui bersama bahwa kecepatan dihitung berdasarkan jarak yang ditempuh sesuatu dalam satu satuan waktu. Berdasarkan prinsip ini, rumus dasar untuk menghitung jarak antara sensor dan benda/penghalang adalah dengan menggunakan rumus sederhana berikut:

$$s = \frac{V \times t}{2}$$

Dengan  $s$  adalah jarak antara sensor dan penghalang,  $V$  adalah kecepatan suara, dan  $t$  adalah waktu antara sinyal dikirim dan diterima. Rumus tersebut memiliki faktor pembagi 2, sebab waktu yang terdeteksi adalah waktu saat sinyal dikirim dan diterima. Waktu ini adalah dua kali waktu untuk mencapai sensor dan penghalang, sehingga jarak yang terdeteksi dengan waktu ini adalah dua kali jarak sensor dan penghalang. Dengan demikian, kita harus menambahkan faktor pembagi dua.

Jarak dihitung berdasarkan **waktu tempuh**, BUKAN berdasarkan intensitas yang diterima. Sudah tentu, intensitas yang sinyal pantul akan berkurang jika dibandingkan dengan sinyal awal.

Kecepatan suara di udara mencapai 331.45 m/s pada temperatur 0 derajat Celcius. Kecepatan suara pada temperatur yang berbeda dapat dihitung dengan rumus berikut:

[rumus kecepatan suara pada temperatur yang berbeda.png](#)

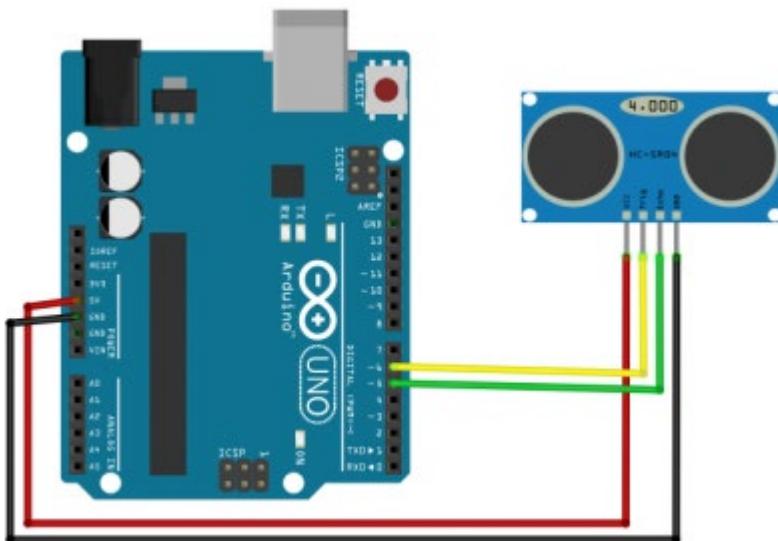
$$V = 331.45 \text{ m/s} + 0.607 \text{ m/s} \times T^{\circ}\text{C}$$

Hal ini dikarenakan kecepatan suara akan meningkat 0.607 m/s setiap kenaikan 1 derajat Celcius.

## Skema Rangkaian

Aturlah sensor sesuai dengan Tabel berikut:

Nama Konektor Sensor	Nomer Pin pada Arduino
Vcc	5V
Gnd	GND
Echo	5
Trig	6



## Pemrograman MATLAB

Untuk MATLAB, ketiklah sintaks **berikut**. Jangan lupa **mengatur COM** sesuai dengan nomer yang terdeteksi pada komputer kita:

```

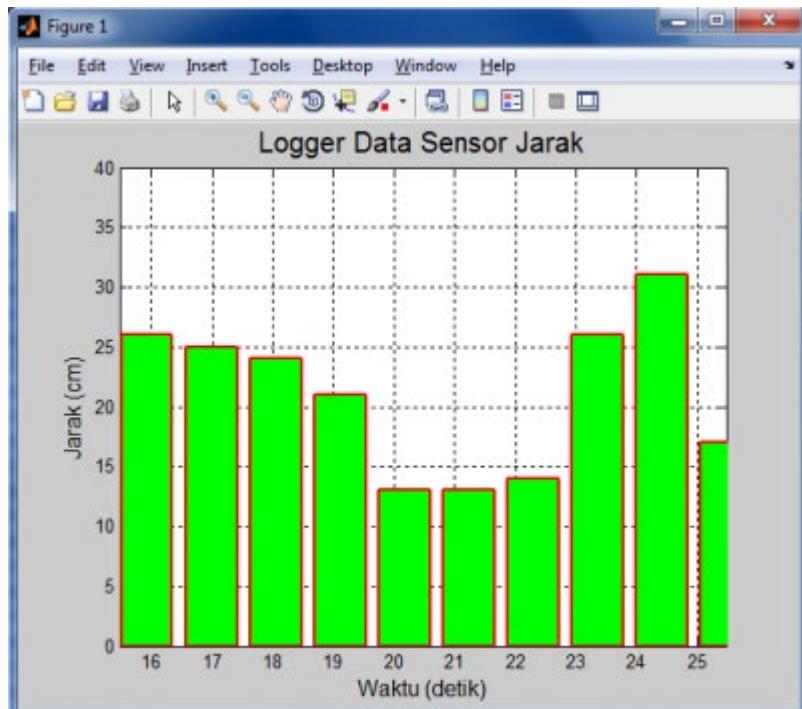
1 clear all;
2 clc
3 %Inisialisasi serial dan grafik
4 serialPort = 'COM33'; % isi dgn COM yg sesuai
5 judulGrafik = 'Logger Data Sensor Jarak'; % Judul grafik
6 xlabel = 'Waktu (detik)'; % x-axis label
7 ylabel = 'Jarak (cm)'; % y-axis label
8 barGrid = 'on'; % aktifkan grid
9 min = 0; % minimum axis-y
10 max = 40; % maksimum axis-y
11 lebarScroll = 10; % display data pada
12 grafik % waktu cuplik
13 delay = 1; %
```

```
13%Inisialisasi variabel
14waktu = 0;
15data = 0;
15cacah = 0;
16
17%Persiapkan grafik
18barGraph = bar(waktu,data,'g',...
19    'LineWidth',2,...
20    'EdgeColor','r');
20title(judulGrafik,'FontSize',15);
21 xlabel(xLabel,'FontSize',12);
22 ylabel(yLabel,'FontSize',12);
23 axis([0 10 min max]);
24 grid(barGrid); %aktifkan grid
25 s = serial(serialPort); %Buka komunikasi melalui port
25 COM
26 set(s,'Timeout',1); % set timeout utk komunikasi
27 disp('Tutup jendela grafik untuk mengakhiri logger');
28 fopen(s); %kirim 1 untuk mengirim
29 fprintf(s,'%s',char(1));
30 Request
30 tic %aktifkan deteksi waktu
31 while ishandle(barGraph) %Terus looping semasih plot
32 aktif
33 nilaiInput = fscanf(s,'%f'); %Baca data serial dalam
34 format float
35 disp('Data Diterima!')
35 %Pastikan data yg diterima benar
36 if(~isempty(nilaiInput) && isfloat(nilaiInput))
37     cacah = cacah + 1;
38     waktu(cacah) = toc; %ambil waktu saat ini
39     data(cacah) = nilaiInput(1); %ambil data saat ini
40
40 %Set Axis sesuai dengan nilai lebarScroll
41 if(lebarScroll > 0)
42     set(barGraph,'XData',waktu(waktu > waktu(cacah)-
43 lebarScroll), ...
44         'YData',data(waktu > waktu(cacah)-lebarScroll));
45     axis([waktu(cacah)-lebarScroll waktu(cacah) min max]);
46 else
46     set(barGraph,'XData',waktu,'YData',data);
47     axis([0 waktu(cacah) min max]);
48 end
49
50 %Beri waktu sesaat utk Update Plot
51 pause(delay);
51 end
52 fprintf(s,'%s',char(1)); %kirim '1' untuk mengirim
53 Request
54 disp('Request Dikirim... ')
55 end
56 % Tutup serial port dan delete variabel yg sudah terpakai
57 fclose(s);
58 clear all;
59 disp('Logger berakhir... ');
60
61
```

62  
63  
64

Amatilah Gambar 3. Gambar tersebut menampilkan contoh hasil yang didapat. Semoga tutorial ini bermanfaat

### Hasil Antarmuka Logger Sensor Ultrasonik.png



**Gambar 3. Hasil Antarmuka Logger Sensor Ultrasonik**

Sumber : <https://tutorkeren.com/artikel/tutorial-ploting-data-real-time-dari-sensor-ultrasonik-menggunakan-matlab.htm>

**SELESAI**

## Distance-Based Similarity Measure

Distance-Based Similarity Measure mengukur tingkat kesamaan dua buah objek dari segi jarak geometris dari variabel-variabel yang tercakup di dalam kedua objek tersebut. Yang termasuk sebagai distance-based similarity measure antara lain:

- **Manhattan Distance (L1 Norm)** : mengukur jarak dua buah objek dengan rumus sebagai berikut:

$$d_{L1}(x_1, x_2) = \text{SUM } (i=0 \text{ to } n) |x_{1i} - x_{2i}|$$

- **Euclidean Distance (L2 Norm)** : mengukur jarak dua buah objek dengan rumus sebagai berikut:

$$d_{L2}(x_1, x_2) = \text{SQRT} ( \text{SUM } (i=0 \text{ to } n) (x_{1i} - x_{2i})^2 )$$

- **Minkowski Distance (Lp Norm)** : mengukur jarak dua buah objek dengan rumus sebagai berikut:

$$d_{Lp}(x_1, x_2) = \text{ROOT}_p ( \text{SUM } (i=0 \text{ to } n) (x_{1i} - x_{2i})^p )$$

- **Chebyshev Distance (Chessboard Distance, L\_Infinity Norm)** : mengukur jarak dua buah objek dengan rumus sebagai berikut:

$$d_{\text{Chebyshev}}(x_1, x_2) = \max_i \{x_{1i} - x_{2i}\}$$

- **Mahalanobis Distance** : mengukur jarak dua buah objek seperti L2 Norm dengan memikirkan korelasi antar objek dalam bentuk vector variabel dari objek dan matrik covariance dari kedua objek tersebut dengan rumus sebagai berikut:

$$d_{\text{Mahalanobis}}(x_1, x_2) = \text{SQRT} ((x_1 - x_2)^T \text{COV}^{-1} (x_1 - x_2))$$

Apabila matrik covariance adalah matrik identity maka Mahalanobis distance adalah Euclidean distance, dan apabila matrik covariance adalah matrik diagonal maka Mahalanobis distance adalah Normalised Euclidean distance dimana korelasi antara objek dianggap tidak ada. Dalam hal ini Mahalanobis distance dihitung dengan rumus sebagai berikut:

$$d_{\text{Mahalanobis}}(x_1, x_2) = \text{SQRT} ( \text{SUM } (i=0 \text{ to } n) (x_{1i} - x_{2i})^2 / \text{SIGMA}_i^2 )$$

- **Hamming Distance** : mengukur jarak antara dua string yang ukurannya sama dengan membandingkan simbol-simbol yang terdapat pada kedua string pada posisi yang sama. Hamming distance dari dua string adalah jumlah simbol dari kedua string yang berbeda. Sebagai contoh Hamming distance antara string ‘toned’ dan ‘roses’ adalah 3. Hamming Distance juga digunakan untuk mengukur jarak antar dua string binary misalnya jarak antara 10011101 dengan 10001001 adalah 2.

- **Levenshtein Distance** : mengukur jarak antara dua string yang ukurannya tidak sama dengan menghitung jumlah pengoperasian yang perlu dilakukan untuk mengubah string yang satu menjadi string yang kedua yang diperbandingkan. Pengoperasian yang dilakukan termasuk operasi insert, delete dan substitusi. Sebagai contoh Levenshtein distance antara string ‘kitten’ dan ‘sitting’ adalah 3 dengan pengoperasian substitusi k dengan s, substitusi e dengan i, dan insert g.
- **Hausdorff Distance** : mengukur jarak berbasis nilai infimum/greatest lower bound dan supremum/greatest upper bound dari kedua objek, dimana semua variabel dari kedua objek tersebut mempunyai nilai compact/closed. Hausdorff distance dihitung dengan rumus sebagai berikut:

$$d_{\text{Hausdorff}}(x_1, x_2) = \max \{ \sup_{x_1 \in x_1} \inf_{x_2 \in x_2} d_{\text{L2}}(x_1, x_2), \sup_{x_2 \in x_2} \inf_{x_1 \in x_1} d_{\text{L2}}(x_1, x_2) \}$$

### Probabilistic-Based Similarity Measure

Probabilistic-Based Similarity Measure menghitung tingkat kemiripan dua objek dengan merepresentasikan dua set objek yang diperbandingkan tersebut dalam bentuk probability. Beberapa metode probabilistic-based similarity measure termasuk:

- **Kullback Leibler Distance** : mengukur tingkat kemiripan variabel objek yang direpresentasikan dalam bentuk probabilitas dua distribusi statistik. Sering disebut juga information distance, information gain, atau relative entropy. Jarak antara dua objek yang bernilai diskrit dalam Kullback Leibler distance dihitung dengan rumus sebagai berikut:

$$D_{\text{KL}}(P, Q) = \sum_i P(i) \log(P(i)/Q(i))$$

Sedang untuk objek yang bernilai continuous dihitung dengan rumus sebagai berikut:

$$D_{\text{KL}}(P, Q) = \int P(i) \log(P(i)/Q(i))$$

- **Posterior Probability** : mengukur tingkat kemiripan dengan mempertimbangkan nilai posterior probability terhadap nilai perbedaan variabel dari kedua objek yang diperbandingkan. Untuk menentukan posterior probability dari perbedaan variabel tersebut diperlukan data nilai-nilai perbedaan yang independent sebagai bahan training untuk pembentukan fungsi likelihood dari perbedaan-perbedaan tersebut.

### Set-Based Similarity Measure

Jaccard Index adalah indeks yang menunjukkan tingkat kesamaan antara suatu himpunan (set) data dengan himpunan (set) data yang lain. Jaccard Index dihitung menggunakan rumus sebagai berikut:

$$J(A, B) = |A \cap B| / (|A| + |B|)$$

Sebagai kebalikannya, tingkat ketidak samaan antara dua himpunan dihitung dengan:

$$J_{\text{delta}}(A,B) = ((A \cup B) - (A \cap B)) / (A \cup B)$$

### **Feature-Based Similarity Measure**

Feature-based similarity measure melakukan penghitungan tingkat kemiripan dengan merepresentasikan objek ke dalam bentuk feature-feature yang ingin diperbandingkan. Feature-based similarity measure banyak digunakan dalam melakukan pengklasifikasian atau pattern matching untuk gambar dan text.

### **Context-Based Similarity Measure**

Context-based similarity measure melakukan penghitungan tingkat kemiripan objek-objek yang mempunyai struktur yang tidak biasa seperti objek yang harus direpresentasikan dengan tree structure atau struktur yang lainnya.

<https://yudiagusta.wordpress.com/2008/05/13/similarity-measure/>