TINGKAT KECANDUAN SESEORANG TERHADAP MAKANAN YANG MENGANDUNG MSG

PENGENALAN POLA - T



Disusun Oleh:

Kelompok 11

Firman Budi Safrizal 155150201111147

Fadhlan Muhammad 155150201111108

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

NOVEMBER 2017

BAB I

LATAR BELAKANG

Tema yang kami pilih pada pembuatan project akhir ini adalah kesehatan. Kami tertarik dengan trend masyarakat Indonesia yang membahas bahwa generasi sekarang (Kids Zaman Now) adalah generasi yang tidak lepas dari micin (MSG). Trend tersebut muncul karena semakin banyaknya generasi muda zaman sekarang sering melakukan hal-hal yang tidak bermanfaat bahkan membahayakan dan terkesan bodoh. Ditambah lagi dengan opini tentang Monosodium Glutamat (MSG) yang sudah menyebar luas di Masyarakat di Indonesia yaitu “Jangan berlebihan konsumsi micin, nanti bikin bodoh”.

BAB II

DEFINISI SISTEM

Judul yang kami ajukan sebagai project akhir Pengenalan Pola adalah “TINGKAT KECANDUAN SESEORANG TERHADAP MAKANAN YANG MENGANDUNG MSG”. Dengan banyaknya makanan yang mengandung MSG, bahkan hampir semua makanan yang dijual umum di Indonesia menggunakan MSG. Hal tersebut membuat kami tertarik untuk melakukan survei apakah seseorang mengalami kecanduan MSG atau tidak.

Monosodium Glutamat (MSG) merupakan sekumpulan Kristal putih kecil yang dikenal mampu menyedapkan rasa suatu makanan. Faktanya, Monosodium Glutamat (MSG) merupakan garam yang molekul penyusunnya berupa sodium atau natrium (Na). Sedangkan garam dapur sendiri berasal dari sodium dan klorida (NaCl). Sehingga perbedaan antara Monosodium Glutamat (MSG) dengan garam dapur terletak pada klorida (Cl) yang digantikan dengan glutamat.

Berdasarkan survei yang dilakukan Persatuan Pabrik Monosodium Glutamat & Asam Glutamat Indonesia (P2MI), konsumsi MSG di Indonesia meningkat dari 100.568 ton pada 1998 menjadi 122.966 ton pada 2004 (diperkirakan 1,53 gram/orang/hari). Angka 1,53 tersebut bisa saja bertambah karena seringnya konsumsi MSG yang tidak disadari. Contohnya pada makanan berkuah atau pada makanan yang tidak mencantumkan kandungan MSG padahal sebenarnya mengandung MSG.

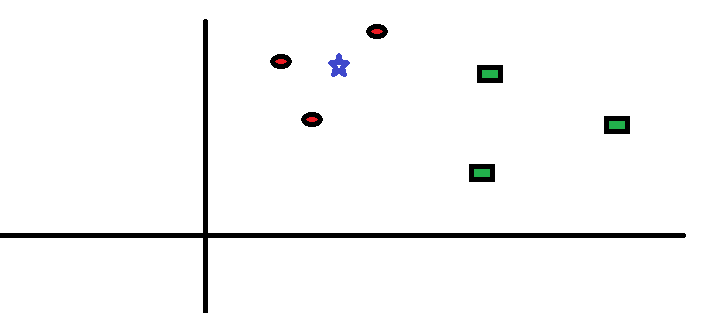
Lalu, apakah MSG berbahaya? Food and Drug Administratrion (FDA)atau badan pengawas obat dan makanan Amerika Serikat mengkategorikan MSG sebagai zat yang cukup aman untuk dikonsumsi. Namun, harus diingat bahwa kata aman tersebut didapat jika dosisnya sesuai dengan rekomendasi mereka. Badan POM Amerika Serikat menyatakan batas aman penggunaan MSG adalah 2 gram/orang/hari. Berbagai laporan mengenai reaksi yang dapat terpicu oleh MSG antara lain tubuh lemas, kulit merah, berkeringat, nyeri dada, sakit kepala, mual.

BAB III

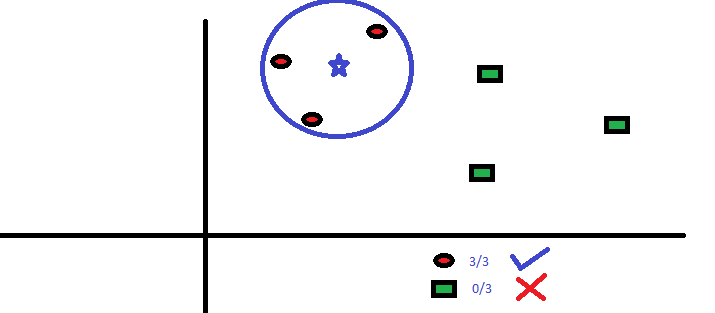
METODE

3.1. Dasar Teori Klasifikasi yang Digunakan

Metode yang kami gunakan untuk mengklasifikasikan data adalah algoritma K-Nearest Neighbor. K-Nearest Neighbour atau K-NN adalah salah dari algoritma instance based learning atau case-based reasoning. Algoritma K-Nearest Neighbor (KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran (data training) yang jaraknya paling dekat dengan objek tersebut. Dengan kata lain membandingkan kemiripan yang paling banyak dalam data pembelajaran (data training). Berikut adalah konsep K-Nearest Neighbor (K-NN):



Ilustrasi 1



Ilustrasi 2

Perhatikan ilustrasi diagram pertama, terdapat lingkaran merah dan persegi panjang hijau di analogikan sebagai kelas, dan sebuah bintang biru dianalogikan sebagai objek yang ingin diklasifikasi. Kita ingin mengklasifikasi, bintang biru tersebut termasuk kelas lingkaran merah atau persegi panjang hijau. Dengan algoritma K-NN, Nilai K adalah tetangga terdekat untuk dilakukan voting. Misal kita ambil K = 3, kita buat lingkaran dengan bintang biru sebagai titik pusat, sebesar mendekati 3 tetangga. Dengan melihat ilustrasi diagram kedua, dapat kita simpulkan bahwa bintang biru tersebut tergolong ke kelas lingkaran merah.

Alasan mengapa kami memilih K-Nearest Neighbor (K-NN) untuk mengklasifikasikan data adalah karena K-Nearest Neighbor nonlinear dan mudah dipahami dan diimplementasikan.

Untuk melakukan pengklasifikasian data dengan menggunakan algoritma K-Nearest Neighbor (K-NN) adalah sebagai berikut:

1. Menentukan parameter k (jumlah tetangga paling dekat).
2. Menghitung distance vector objek terhadap data training yang diberikan.
3. Mengurutkan hasil no 2 secara ascending (berurutan dari nilai rendah ke tinggi).
4. Mengumpulkan kategori Y (Klasifikasi nearest neighbor berdasarkan nilai k).
5. Dengan menggunakan kategori nearest neighbor yang paling mayoritas maka dapat diprediksikan kategori objek.

Sebagai contoh terdapat data sebagai berikut:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data | Fitur 1 | Fitur 2 | Fitur 3 | Kelas |
| 1 | 10 | 30 | 10 | A |
| 2 | 20 | 10 | 10 | A |
| 3 | 30 | 20 | 15 | A |
| 4 | 5 | 2 | 2 | B |
| 5 | 4 | 8 | 3 | B |
| 6 | 10 | 10 | 10 | ? |

**Langkah 1 : Menentukan parameter k (jumlah tetangga paling dekat).**

Untuk menentukan K, terdapat dua cara, yakni unweighted voting dan weighted voting.

* Weighted Voting adalah menentukan langsung nilai K. Contoh K = 3.
* Unweighted voting adalah menentukan nilai K dengan menghitung jumlah distance terhadap semua data.

Weight Kelas A :

Weight Kelas B :

Lalu Diambil nilai Weight terbesar diantara kedua kelas tersbeut. Dan klasifikasi kelas mengikuti nilai Weight terbesar.

Pada kasus ini kita gunakan weighted voting dengan nilai K = 3.

**Langkah 2: Menghitung distance vector objek terhadap data training yang diberikan.**

Untuk menghitung distance vector dapat menggunakan Euclidean Distance, yaitu:

Keterangan :

x, y, z : data uji

xi,yi,zi : data training ke-i

n : hingga sebanyak jumlah fitur

Berikut adalah perhitungan Distance dengan Euclidean Distance dari data table diatas.

|  |  |
| --- | --- |
| Data | Distance |
| 6 🡪 1 | (10-10)­2 + (10-30)­2 + (10-10)­2 = 20 |
| 6 🡪 2 | (10-20)­2 + (10-10)­2 + (10-10)­2 = 10 |
| 6 🡪 3 | (10-30)­2 + (10-20)­2 + (10-15)­2 = 22,9 |
| 6 🡪 4 | (10-5)­2 + (10-2)­2 + (10-2)­2 = 12,36 |
| 6 🡪 5 | (10-4)­2 + (10-8)­2 + (10-3)­2 = 9,4 |

**Langkah 3 : Mengurutkan hasil no 2 secara ascending (berurutan dari nilai rendah ke tinggi).**

Dari yang diperoleh dari langkah 2 kita urutkan dari yang terkecil.

|  |  |  |
| --- | --- | --- |
| Urutan Tetangga Terdekat | Distance | Kelas |
| 5 | 9,4 | B |
| 2 | 10 | A |
| 4 | 12,36 | B |
| 1 | 20 | A |
| 3 | 22,9 | A |

**Langkah 4 : Mengumpulkan kategori Y (Klasifikasi nearest neighbor berdasarkan nilai k).**

Berdasarkan nilai K yang sudah ditentukan pada langkah pertama, yaitu K = 3. Lalu kita ambil 3 data tetangga terdekat.

|  |  |  |
| --- | --- | --- |
| Urutan Tetangga Terdekat | Distance | Kelas |
| 5 | 9,4 | B |
| 2 | 10 | A |
| 4 | 12,36 | B |

**Langkah 5 : Dengan menggunakan kategori nearest neighbor yang paling mayoritas maka dapat diprediksikan kategori objek.**

Dari 3 data pada langkah 4, terdapat 2 kelas B dan 1 kelas A. Karena jumlah kelas B lebih banyak dari jumlah kelas A. Maka data ke-6 tadi dapat kita klasifikasikan sebagai **kelas B.**

3.2. Proses Training

Untuk mengklasifikasikan project akhir kami yang berjudul “Tingkat Kecanduan Seseorang Terhadap Makanan yang Mengandung MSG”. Kami menggunakan tiga Fitur dan tiga Tingkatan Kelas.

Fitur :

1. Rata-rata Jumlah Makanan Mengandung MSG Yang Anda Konsumsi Dalam 1 Hari
2. Rata-rata Jumlah Makanan Tidak Mengandung MSG Yang Anda Konsumsi Dalam 1 Hari
3. Body Mass Index (BMI)

Berikut adalah alasan mengapa kami memilih ketiga fitur diatas untuk mengklasifikasikan seseorang terdapat kecenderungan untuk konsumsi Monosodium Glutamat (MSG) secara berlebih.

Pertama, kami melihat dari banyaknya rata-rata jumlah makanan mengandung MSG yang dikonsumsi dalam sehari. Dengan jumlah tersebut, kami dapat mengetahui bahwa seseorang berlebihan atau tidak dalam konsumsi MSG dalam sehari.

Kedua, kami melihat dari banyaknya rata-rata jumlah makanan tidak mengandung MSG yang dikonsumsi dalam sehari. Fitur ini kami perlukan untuk melihat keseimbangan antara makanan mengandung MSG dan makanan tidak mengandung MSG. Jika jumlah makanan yang mengandung MSG lebih banyak daripada yang tidak, maka terdapat kecenderungan bahwa orang tersebut kecanduan MSG.

Ketiga, kami melihat dari Body Mass Index (BMI). BMI sendiri merupakan ukuran yang digunakan untuk menilai proporsionalitas perbandingan antara tinggi dan berat seseorang. BMI sering digunakan dokter untuk menilai seseorang itu obesitas atau tidak. Kami menggunakan BMI dikarenakan beberapa penelitian mengklaim, bahwa bahan MSG memengaruhi pankreas dan kadar gula darah. Secara tidak langsung dapat membuat seseorang makan lebih banyak dan dengan demikian menyebabkan kegemukan. Untuk mendapatkan nilai BMI, kita dapat menggunakan rumus berikut:

BMI = Berat Badan (kg) / Tinggi Badan (m) \* Tinggi Badan (m)

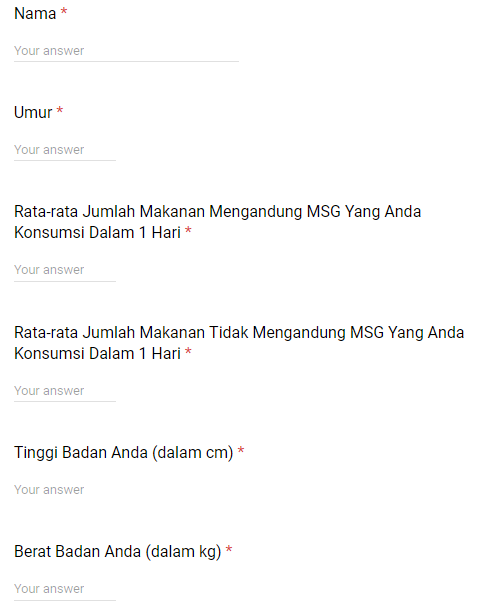
Setelah mendapatkan nilai BMI, nilai tersebut kami kategorikan menjadi empat kategori, sebagai berikut :

|  |  |  |
| --- | --- | --- |
| Nilai BMI | Kategori Berat Badan | Kode Kategori |
| <= 18.5 | Kurang | 0 |
| 18.5 < x <= 25 | Ideal | 1 |
| 25 < x <= 30 | Kelebihan | 2 |
| > 30 | Obesitas | 3 |

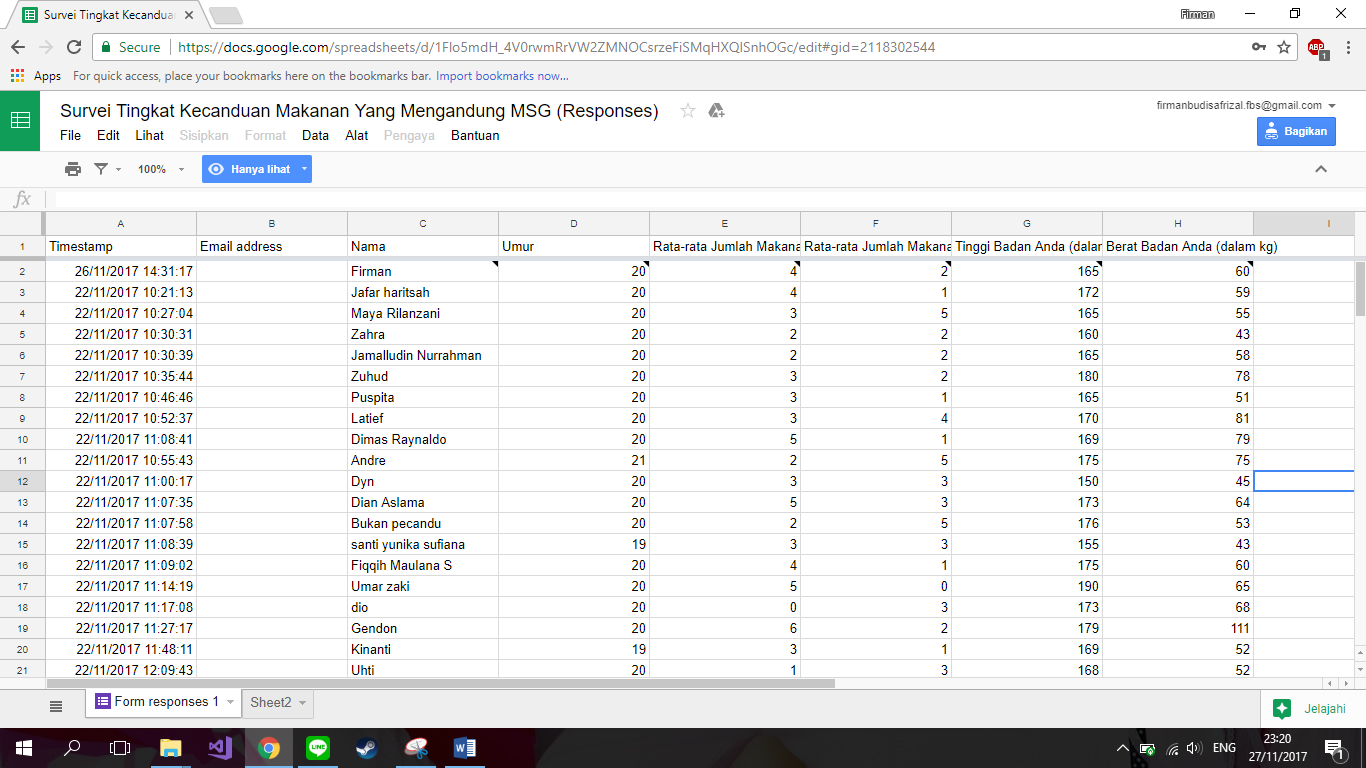
Tingkatan Kelas :

1. Sehat, kami gunakan untuk mengklasifikasikan orang yang mengkonsumsi sedikit MSG dan menjalani hidup sehat.
2. Cukup, kami gunakan untuk mengklasifikasikan orang yang seimbang dalam mengkonsumsi makanan mengandung MSG dan yang tidak mengandung MSG.
3. Pecandu, kami gunakan untuk mengklasifikasikan orang yang mengkonsumsi MSG secara berlebihan.

Dalam mengerjakan project akhir ini, kami mencari data primer dengan mensurvei orang dewasa yang berumur 19-21. Dalam survei tersebut kami menggunakan bantuan dari Google Forms. Berikut adalah contoh form yang digunakan untuk survei.



Kemudian kita dapat melihat hasil data primer tersebut melalui google spreadsheets. Lalu kemudian data tersebut kami olah untuk mendapatkan nilai BMI



Data tersebut kami olah untuk mendapatkan kategori BMI dan mengklasifikasi data ke tingkatan kelas. Untuk menilai bahwa seseorang tergolong kelas sehat, cukup, maupun pecandu, data tersebut kami tanyakan kepada orang lain agar penilaian dapat objektif. Berikut adalah data yang telah diolah :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data Ke | Rata-rata Jumlah Makanan Mengandung MSG Yang Anda Konsumsi Dalam 1 Hari | Rata-rata Jumlah Makanan Tidak Mengandung MSG Yang Anda Konsumsi Dalam 1 Hari | Kategori BMI | Kelas |
| 1 | 6 | 2 | 1 | Pecandu |
| 2 | 4 | 1 | 1 | Pecandu |
| 3 | 3 | 5 | 1 | Sehat |
| 4 | 2 | 2 | 0 | Sehat |
| 5 | 2 | 2 | 1 | Sehat |
| 6 | 3 | 2 | 1 | Sehat |
| 7 | 3 | 1 | 1 | Sehat |
| 8 | 4 | 1 | 2 | Pecandu |
| 9 | 5 | 1 | 2 | Pecandu |
| 10 | 2 | 5 | 1 | Cukup |
| 11 | 3 | 3 | 1 | Cukup |
| 12 | 5 | 3 | 1 | Pecandu |
| 13 | 3 | 3 | 0 | Sehat |
| 14 | 4 | 1 | 1 | Pecandu |
| 15 | 5 | 0 | 0 | Pecandu |
| 16 | 0 | 3 | 1 | Sehat |
| 17 | 6 | 2 | 3 | Pecandu |
| 18 | 3 | 1 | 0 | Cukup |
| 19 | 1 | 3 | 0 | Sehat |
| 20 | 4 | 2 | 1 | Cukup |
| 21 | 1 | 2 | 1 | Sehat |
| 22 | 4 | 1 | 3 | Pecandu |
| 23 | 4 | 2 | 0 | Pecandu |
| 24 | 2 | 4 | 0 | Sehat |
| 25 | 7 | 0 | 2 | Pecandu |
| 26 | 3 | 2 | 0 | Cukup |
| 27 | 2 | 3 | 1 | Cukup |
| 28 | 3 | 1 | 1 | Cukup |
| 29 | 2 | 3 | 1 | Cukup |
| 30 | 1 | 4 | 1 | Sehat |
| 31 | 2 | 1 | 1 | Cukup |
| 32 | 4 | 5 | 2 | Pecandu |
| 33 | 2 | 5 | 1 | Sehat |
| 34 | 4 | 2 | 0 | Pecandu |
| 35 | 5 | 3 | 1 | Cukup |
| 36 | 4 | 4 | 1 | Cukup |
| 37 | 3 | 2 | 1 | Cukup |
| 38 | 6 | 3 | 2 | Pecandu |
| 39 | 3 | 3 | 1 | Cukup |
| 40 | 5 | 2 | 2 | Pecandu |
| 41 | 2 | 2 | 1 | Sehat |
| 42 | 3 | 2 | 1 | Cukup |
| 43 | 4 | 3 | 1 | Cukup |
| 44 | 5 | 3 | 2 | Pecandu |
| 45 | 1 | 3 | 1 | Sehat |
| 46 | 3 | 3 | 1 | Cukup |
| 47 | 5 | 3 | 2 | Pecandu |
| 48 | 4 | 2 | 1 | Pecandu |
| 49 | 6 | 1 | 3 | Pecandu |
| 50 | 3 | 3 | 2 | Cukup |

Keterangan Kategori BMI :

0 = Kurang

1 = Ideal

2 = Kelebihan

3 = Obesitas

Dari data tersebut, kami gunakan 35 data pertama sebagai data latih, dan 15 data berikutnya sebagai data uji.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Proses Testing

Pada proses testing ini kami menggunakan bantuan program java yang kami coding sendiri. Dengan catatan, kami melakukan testing dengan menggunakan weighted voting K-NN.

Berikut adalah proses pembuatan program tersebut :

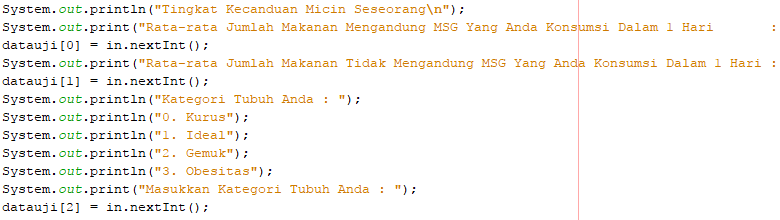
1. Kami buat variable array 1 dimensi dengan panjang 3 bernama datauji untuk menyimpan inputan dari user untuk dilakukan klasifikasi.

datauji[0] = jumlah makanan yang mengandung MSG.

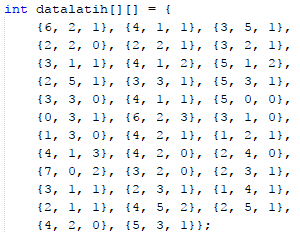
datauji[1] = jumlah makanan yang tidak mengandung MSG.

datauji[2] = kategori BMI.

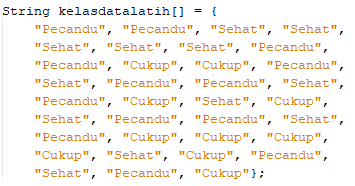




1. Kami buat variable array 2 dimensi dengan panjang 35 bernama datalatih untuk menyimpan data latih berdasarkan tabel data primer yang telah diolah.

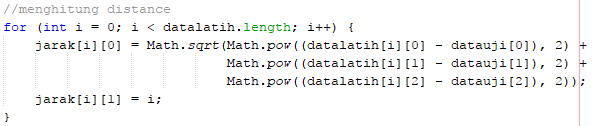


1. Kami buat variable array 1 dimensi dengan panjang 35 bernama kelasdatalatih untuk menyimpan kelas data latih berdasarkan tabel data primer yang telah diolah.

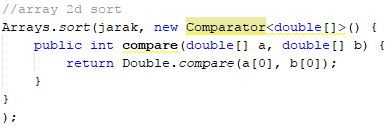


1. Kami buat variable array 2 dimensi dengan panjang 35 bernama jarak untuk menyimpan indeks dan jarak yang dihitung dengan Euclidean Distance.

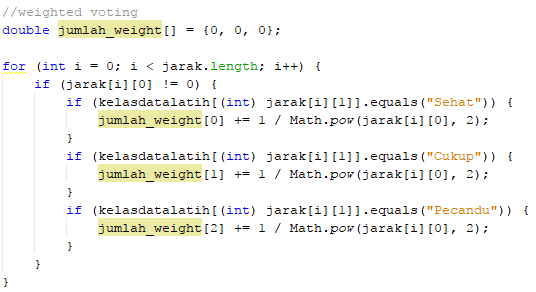




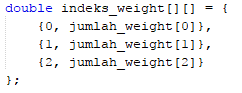
1. Kami urutkan variable jarak secara ascending (kecil ke besar).



1. Karena kami menggunakan weighted voting untuk menentukan kelas, maka kami buat variable array 1 dimensi bernama untuk menyimpan jumlah weight pada tiap kelas sehat, cukup, dan pecandu.

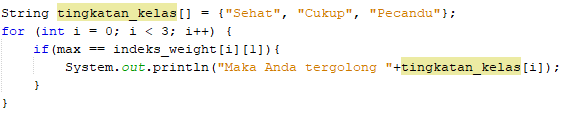


1. Untuk mendapatkan nilai tertinggi dari weight sehat, pertama kami buat variable array 2 dimensi untuk menyimpan indeks, dan jumlah weight dari tiap kelas. Kemudian kami buat variable max untuk menyimpan jumlah weight tertinggi.





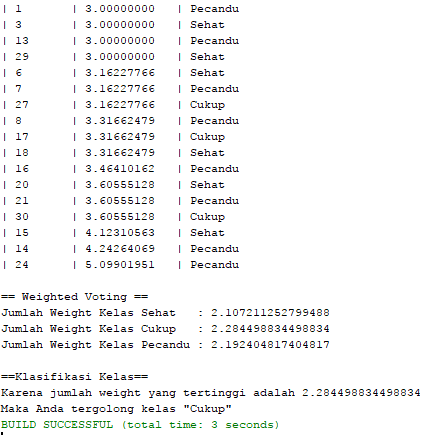
1. Untuk mendapatkan klasifikasi tingkatan kelas berdasarkan nilai weight tertinggi, pertama kami buat variable tingkatan kelas. Kemudian kami kita cek, jika nilai max sama dengan nilai weight pada variable indeks\_weight maka, data tersebut tergolong tingkatankelas[i].



1. Sehingga jika codingan tersebut digabung akan menjadi seperti berikut:

|  |  |
| --- | --- |
|  | ProjectAkhir.java |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139 | package javaapplication4;  import java.util.\*;  import java.util.Scanner;  /\*\*  \*  \* @author frmnjn & fadhlanmuhammad  \*/  public class Project\_Akhir {  public static void main(String[] args) {  Scanner in = new Scanner(System.in);    int datauji[] = new int[3];    int datalatih[][] = {  {6, 2, 1}, {4, 1, 1}, {3, 5, 1},  {2, 2, 0}, {2, 2, 1}, {3, 2, 1},  {3, 1, 1}, {4, 1, 2}, {5, 1, 2},  {2, 5, 1}, {3, 3, 1}, {5, 3, 1},  {3, 3, 0}, {4, 1, 1}, {5, 0, 0},  {0, 3, 1}, {6, 2, 3}, {3, 1, 0},  {1, 3, 0}, {4, 2, 1}, {1, 2, 1},  {4, 1, 3}, {4, 2, 0}, {2, 4, 0},  {7, 0, 2}, {3, 2, 0}, {2, 3, 1},  {3, 1, 1}, {2, 3, 1}, {1, 4, 1},  {2, 1, 1}, {4, 5, 2}, {2, 5, 1},  {4, 2, 0}, {5, 3, 1}};    String kelasdatalatih[] = {  "Pecandu", "Pecandu", "Sehat", "Sehat",  "Sehat", "Sehat", "Sehat", "Pecandu",  "Pecandu", "Cukup", "Cukup", "Pecandu",  "Sehat", "Pecandu", "Pecandu", "Sehat",  "Pecandu", "Cukup", "Sehat", "Cukup",  "Sehat", "Pecandu", "Pecandu", "Sehat",  "Pecandu", "Cukup", "Cukup", "Cukup",  "Cukup", "Sehat", "Cukup", "Pecandu",  "Sehat", "Pecandu", "Cukup"};    double jarak[][] = new double[datalatih.length][2];  System.out.println("Tingkat Kecanduan Micin Seseorang\n");  System.out.print("Rata-rata Jumlah Makanan Mengandung MSG Yang Anda Konsumsi Dalam 1 Hari : ");  datauji[0] = in.nextInt();  System.out.print("Rata-rata Jumlah Makanan Tidak Mengandung MSG Yang Anda Konsumsi Dalam 1 Hari : ");  datauji[1] = in.nextInt();  System.out.println("Kategori Tubuh Anda : ");  System.out.println("0. Kurus");  System.out.println("1. Ideal");  System.out.println("2. Gemuk");  System.out.println("3. Obesitas");  System.out.print("Masukkan Kategori Tubuh Anda : ");  datauji[2] = in.nextInt();        System.out.println("\n== Data Distance Ascending ==");    //menghitung distance  for (int i = 0; i < datalatih.length; i++) {  jarak[i][0] = Math.sqrt(Math.pow((datalatih[i][0] - datauji[0]), 2) +  Math.pow((datalatih[i][1] - datauji[1]), 2) +  Math.pow((datalatih[i][2] - datauji[2]), 2));  jarak[i][1] = i;  }  //array 2d sort  Arrays.sort(jarak, new Comparator<double[]>() {  public int compare(double[] a, double[] b) {  return Double.compare(a[0], b[0]);  }  }  );  System.out.printf("| %s | %s\t | %s \n","Data Ke","Distance","Kelas");  for (int i = 0; i < datalatih.length; i++) {  //System.out.println(jarak[i][0] + "\t\t\t" + kelasdatalatih[(int) jarak[i][1]]);  System.out.printf("| %d\t | %.8f\t | %s \n",(int)jarak[i][1],jarak[i][0],kelasdatalatih[(int)jarak[i][1]]);  }  System.out.println("\n== Weighted Voting ==");      //weighted voting  double jumlah\_weight[] = {0, 0, 0};  for (int i = 0; i < jarak.length; i++) {  if (jarak[i][0] != 0) {  if (kelasdatalatih[(int) jarak[i][1]].equals("Sehat")) {  jumlah\_weight[0] += 1 / Math.pow(jarak[i][0], 2);  }  if (kelasdatalatih[(int) jarak[i][1]].equals("Cukup")) {  jumlah\_weight[1] += 1 / Math.pow(jarak[i][0], 2);  }  if (kelasdatalatih[(int) jarak[i][1]].equals("Pecandu")) {  jumlah\_weight[2] += 1 / Math.pow(jarak[i][0], 2);  }  }  }    System.out.println("Jumlah Weight Kelas Sehat : " + jumlah\_weight[0]);  System.out.println("Jumlah Weight Kelas Cukup : " + jumlah\_weight[1]);  System.out.println("Jumlah Weight Kelas Pecandu : " + jumlah\_weight[2]);    double indeks\_weight[][] = {  {0, jumlah\_weight[0]},  {1, jumlah\_weight[1]},  {2, jumlah\_weight[2]}  };  double max = Math.max(Math.max(jumlah\_weight[0], jumlah\_weight[1]), jumlah\_weight[2]);    String tingkatan\_kelas[] = {"Sehat", "Cukup", "Pecandu"};  System.out.println("\n==Klasifikasi Kelas==");  System.out.println("Karena jumlah weight yang tertinggi adalah "+max);  for (int i = 0; i < 3; i++) {  if(max == indeks\_weight[i][1]){  System.out.println("Maka Anda tergolong kelas \""+tingkatan\_kelas[i]+ "\"");  }  }  }  } |

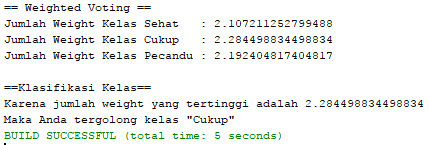




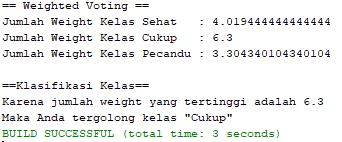
Data yang digunakan dalam proses testing berjumlah 15 data. Namun, program tersebut hanya dapat mengklasifikasikan satu data saja. Sehingga untuk mengklasifikasikan 15 data, maka kita harus menjalankan program sebanyak 15 kali juga untuk mendapatkan hasil masing-masing data.

Format Hasil : Data Ke-n (jumlah makanan mengandung MSG, jumlah makanan tidak mengandung MSG, kategori BMI) 🡪 Kelas.

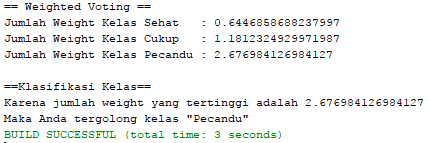
* Data Ke-36 (4, 4, 1) 🡪 Cukup



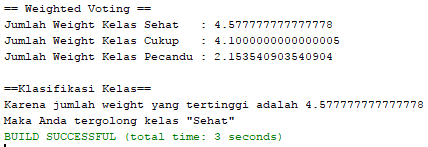
* Data Ke-37 (3, 2, 1) 🡪 Cukup



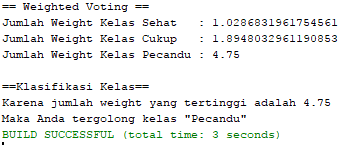
* Data Ke-38 (6, 3, 2) 🡪 Pecandu



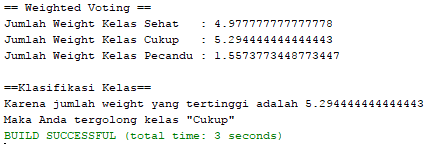
* Data Ke-39 (3, 3, 1) 🡪 Sehat



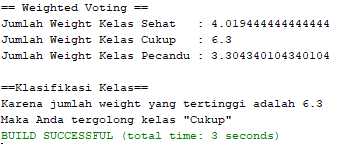
* Data Ke-40 (5, 2, 2) 🡪 Pecandu



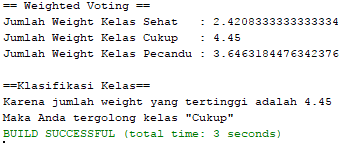
* Data Ke-41 (2, 2, 1) 🡪 Cukup



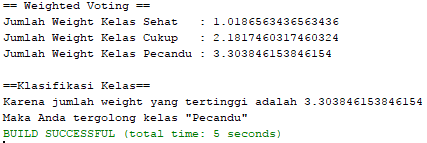
* Data Ke-42 (3, 2, 1) 🡪 Cukup



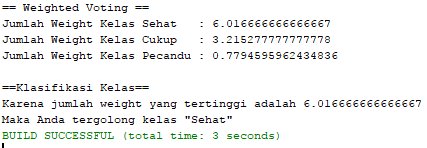
* Data Ke-43 (4, 3, 1) 🡪 Cukup



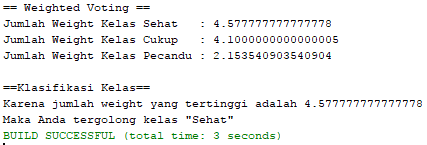
* Data Ke-44 (5, 3, 2) 🡪 Pecandu



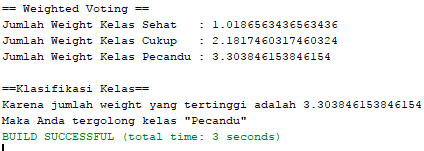
* Data Ke-45 (1, 3, 1) 🡪 Sehat



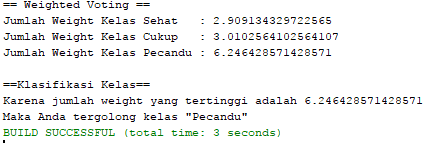
* Data Ke-46 (3, 3, 1) 🡪 Sehat



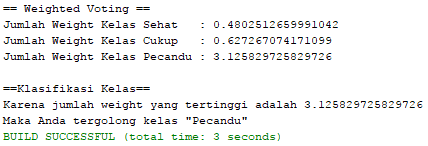
* Data Ke-47 (5, 3, 2) 🡪 Pecandu



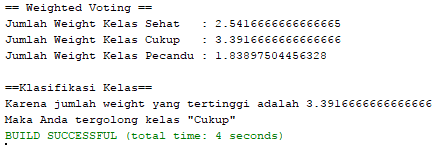
* Data Ke-48 (4, 2, 1) 🡪 Pecandu



* Data Ke-49 (6, 1, 3) 🡪 Pecandu



* Data Ke-50 (3, 3, 2) 🡪 Cukup



Salah Satu contoh hitungan manual pada data ke-36 (4, 4, 1)

Distance Vector dengan menggunakan Euclidean Distance

|  |  |  |
| --- | --- | --- |
| Data Ke | Distance | Kelas |
| 36 --> 1 | (4-6)­2 + (4-2)­2 + (1-1)­2 = 2,82842712474619 | Pecandu |
| 36 --> 2 | (4-4)­2 + (4-1)­2 + (1-1)­2 = 3,00000000000000 | Pecandu |
| 36 --> 3 | (4-3)­2 + (4-5)­2 + (1-1)­2 = 1,41421356237310 | Sehat |
| 36 --> 4 | (4-2)­2 + (4-2)­2 + (1-0)­2 = 3,00000000000000 | Sehat |
| 36 --> 5 | (4-2)­2 + (4-2)­2 + (1-1)­2 = 2,82842712474619 | Sehat |
| 36 --> 6 | (4-3)­2 + (4-2)­2 + (1-1)­2 = 2,23606797749979 | Sehat |
| 36 --> 7 | (4-3)­2 + (4-1)­2 + (1-1)­2 = 3,16227766016838 | Sehat |
| 36 --> 8 | (4-4)­2 + (4-1)­2 + (1-2)­2 = 3,16227766016838 | Pecandu |
| 36 --> 9 | (4-5)­2 + (4-1)­2 + (1-2)­2 = 3,31662479035540 | Pecandu |
| 36 --> 10 | (4-2)­2 + (4-5)­2 + (1-1)­2 = 2,23606797749979 | Cukup |
| 36 --> 11 | (4-3)­2 + (4-3)­2 + (1-1)­2 = 1,41421356237310 | Cukup |
| 36 --> 12 | (4-5)­2 + (4-3)­2 + (1-1)­2 = 1,41421356237310 | Pecandu |
| 36 --> 13 | (4-3)­2 + (4-3)­2 + (1-0)­2 = 1,73205080756888 | Sehat |
| 36 --> 14 | (4-4)­2 + (4-1)­2 + (1-1)­2 = 3,00000000000000 | Pecandu |
| 36 --> 15 | (4-5)­2 + (4-0)­2 + (1-0)­2 = 4,24264068711928 | Pecandu |
| 36 --> 16 | (4-0)­2 + (4-3)­2 + (1-1)­2 = 4,12310562561766 | Sehat |
| 36 --> 17 | (4-6)­2 + (4-2)­2 + (1-3)­2 = 3,46410161513775 | Pecandu |
| 36 --> 18 | (4-3)­2 + (4-1)­2 + (1-0)­2 = 3,31662479035540 | Cukup |
| 36 --> 19 | (4-1)­2 + (4-3)­2 + (1-0)­2 = 3,31662479035540 | Sehat |
| 36 --> 20 | (4-4)­2 + (4-2)­2 + (1-1)­2 = 2,00000000000000 | Cukup |
| 36 --> 21 | (4-1)­2 + (4-2)­2 + (1-1)­2 = 3,60555127546399 | Sehat |
| 36 --> 22 | (4-4)­2 + (4-1)­2 + (1-3)­2 = 3,60555127546399 | Pecandu |
| 36 --> 23 | (4-4)­2 + (4-2)­2 + (1-0)­2 = 2,23606797749979 | Pecandu |
| 36 --> 24 | (4-2)­2 + (4-4)­2 + (1-0)­2 = 2,23606797749979 | Sehat |
| 36 --> 25 | (4-7)­2 + (4-0)­2 + (1-2)­2 = 5,09901951359278 | Pecandu |
| 36 --> 26 | (4-3)­2 + (4-2)­2 + (1-0)­2 = 2,44948974278318 | Cukup |
| 36 --> 27 | (4-2)­2 + (4-3)­2 + (1-1)­2 = 2,23606797749979 | Cukup |
| 36 --> 28 | (4-3)­2 + (4-1)­2 + (1-1)­2 = 3,16227766016838 | Cukup |
| 36 --> 29 | (4-2)­2 + (4-3)­2 + (1-1)­2 = 2,23606797749979 | Cukup |
| 36 --> 30 | (4-1)­2 + (4-4)­2 + (1-1)­2 = 3,00000000000000 | Sehat |
| 36 --> 31 | (4-2)­2 + (4-1)­2 + (1-1)­2 = 3,60555127546399 | Cukup |
| 36 --> 32 | (4-4)­2 + (4-5)­2 + (1-2)­2 = 1,41421356237310 | Pecandu |
| 36 --> 33 | (4-2)­2 + (4-5)­2 + (1-1)­2 = 2,23606797749979 | Sehat |
| 36 --> 34 | (4-4)­2 + (4-2)­2 + (1-0)­2 = 2,23606797749979 | Pecandu |
| 36 --> 35 | (4-5)­2 + (4-3)­2 + (1-1)­2 = 1,41421356237310 | Cukup |

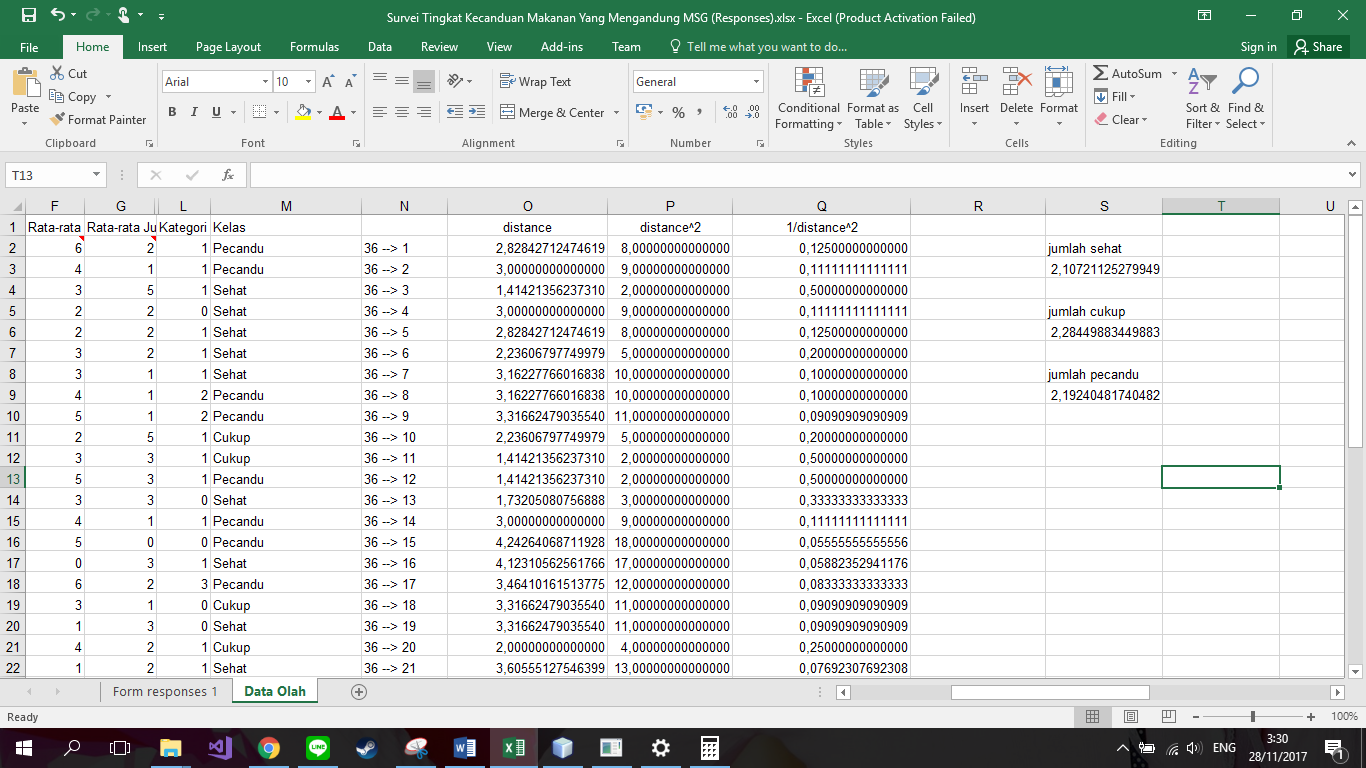
Distance yang sudah di sort berdasarkan distance terkecil hingga terbesar

|  |  |  |
| --- | --- | --- |
| Data Ke | Distance | Kelas |
| 36 --> 3 | 1,41421356237310 | Sehat |
| 36 --> 11 | 1,41421356237310 | Cukup |
| 36 --> 12 | 1,41421356237310 | Pecandu |
| 36 --> 32 | 1,41421356237310 | Pecandu |
| 36 --> 35 | 1,41421356237310 | Cukup |
| 36 --> 13 | 1,73205080756888 | Sehat |
| 36 --> 20 | 2,00000000000000 | Cukup |
| 36 --> 6 | 2,23606797749979 | Sehat |
| 36 --> 10 | 2,23606797749979 | Cukup |
| 36 --> 23 | 2,23606797749979 | Pecandu |
| 36 --> 24 | 2,23606797749979 | Sehat |
| 36 --> 27 | 2,23606797749979 | Cukup |
| 36 --> 29 | 2,23606797749979 | Cukup |
| 36 --> 33 | 2,23606797749979 | Sehat |
| 36 --> 34 | 2,23606797749979 | Pecandu |
| 36 --> 26 | 2,44948974278318 | Cukup |
| 36 --> 1 | 2,82842712474619 | Pecandu |
| 36 --> 5 | 2,82842712474619 | Sehat |
| 36 --> 2 | 3,00000000000000 | Pecandu |
| 36 --> 4 | 3,00000000000000 | Sehat |
| 36 --> 14 | 3,00000000000000 | Pecandu |
| 36 --> 30 | 3,00000000000000 | Sehat |
| 36 --> 7 | 3,16227766016838 | Sehat |
| 36 --> 8 | 3,16227766016838 | Pecandu |
| 36 --> 28 | 3,16227766016838 | Cukup |
| 36 --> 9 | 3,31662479035540 | Pecandu |
| 36 --> 18 | 3,31662479035540 | Cukup |
| 36 --> 19 | 3,31662479035540 | Sehat |
| 36 --> 17 | 3,46410161513775 | Pecandu |
| 36 --> 21 | 3,60555127546399 | Sehat |
| 36 --> 22 | 3,60555127546399 | Pecandu |
| 36 --> 31 | 3,60555127546399 | Cukup |
| 36 --> 16 | 4,12310562561766 | Sehat |
| 36 --> 15 | 4,24264068711928 | Pecandu |
| 36 --> 25 | 5,09901951359278 | Pecandu |

Weighted Voting

Rumus untuk menghitung weight suatu kelas n adalah

Disini kami menggunakan bantuan Microsoft Excel untuk melakukan perhitungan weight kelas.



* **Weight Kelas Sehat :**

Dengan gambar diatas, pada Cell S3, dengan memasukkan formula =SUMIF(M2:M36;"Sehat";Q2:Q36), maka akan menjumlahkan kolom Q2 hingga Q36 apabila pada kolom M berisikan kata “Sehat”. Sehingga didapat hasil 2,10721125279949

* **Weight Kelas Cukup :**

Dengan gambar diatas, pada Cell S3, dengan memasukkan formula =SUMIF(M2:M36;"Cukup";Q2:Q36), maka akan menjumlahkan kolom Q2 hingga Q36 apabila pada kolom M berisikan kata “Cukup”. Sehingga didapat hasil 2,28449883449883

* **Weight Kelas Pecandu :**

Dengan gambar diatas, pada Cell S3, dengan memasukkan formula =SUMIF(M2:M36;"Pecandu";Q2:Q36), maka akan menjumlahkan kolom Q2 hingga Q36 apabila pada kolom M berisikan kata “Pecandu”. Sehingga didapat hasil 2,19240481740482

Dikarenakan WCukup > WPecandu > WSehat. Maka Data Ke 36 dapat digolongkan menjadi kelas “Cukup”.

Akurasi Klasifikasi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data Ke | Fitur 1 | Fitur 2 | Fitur 3 | Prediksi Awal Kelas | Hasil Akhir Testing |
| 36 | 4 | 4 | 1 | Cukup | Cukup |
| 37 | 3 | 2 | 1 | Cukup | Cukup |
| 38 | 6 | 3 | 2 | Pecandu | Pecandu |
| 39 | 3 | 3 | 1 | Cukup | Sehat |
| 40 | 5 | 2 | 2 | Pecandu | Pecandu |
| 41 | 2 | 2 | 1 | Sehat | Cukup |
| 42 | 3 | 2 | 1 | Cukup | Cukup |
| 43 | 4 | 3 | 1 | Cukup | Cukup |
| 44 | 5 | 3 | 2 | Pecandu | Pecandu |
| 45 | 1 | 3 | 1 | Sehat | Sehat |
| 46 | 3 | 3 | 1 | Cukup | Sehat |
| 47 | 5 | 3 | 2 | Pecandu | Pecandu |
| 48 | 4 | 2 | 1 | Pecandu | Pecandu |
| 49 | 6 | 1 | 3 | Pecandu | Pecandu |
| 50 | 3 | 3 | 2 | Cukup | Cukup |

Untuk menghitung prosentase akurasi dapat menggunakan rumus berikut:

Berdasarkan tabel diatas, didapatkan 12 klasifikasi benar dari 15 data uji.

4.2. Pembahasan

Setelah melakukan klasifikasi berdasarkan data primer menggunakan algoritma K-Nearest Neighbor (K-NN), dapat disimpulkan bahwa algoritma K-Nearest Neighbor dapat mengklasifikasi suatu objek. Dalam kasus ini adalah, tingkat kecanduan Seseorang Terhadap Makanan Mengandung MSG.

Dengan diberikan 35 data training, dan diuji dengan 15 data uji, hasil pengklasifikasian objek ini dapat dikatakan berhasil. Karena dari 15 data uji tersebut, program dengan menggunakan algoritma K-Nearest Neighbor ini dapat mengklasifikan 12 data secara benar dan 3 data salah dan didapatkan nilai akurasi 80%.