

**k-nearest neighbor (KNN)**

**Modul 5 Praktikum**

**Statistika Sains Data**

**Program Studi Sains Data**

**Fakultas Sains**

**Institut Teknologi Sumatera**

**2024**

**Eksplorasi Data**

**Modul Praktikum 1**

**Statistika Sains Data**

**Eksplorasi Data**

**Modul Praktikum 1**

**Statistika Sains Data**

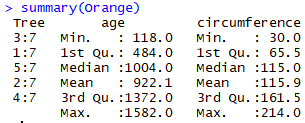
1. **Tujuan Praktikum**
2. Mahasiswa mampu mengklasifikasikan objek baru berdasarkan atribut dan training samples menggunakan algoritma KNN.
3. Mahasiswa mampu mengimplementasi metode K-nearest neighbor (KNN).
4. **Teori Dasar**

Algoritma k-nearest neighbor (KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut. KNN termasuk algoritma supervised learning dimana hasil dari query instance yang baru diklasifikan berdasarkan mayoritas dari kategori pada KNN. Nanti kelas yang paling banyak muncullah yang akan menjadi kelas hasil klasifikasi.

Tujuan dari algoritma ini adalah mengklasifikasikan obyek baru bedasarkan atribut dan training sample. Classifier tidak menggunakan model apapun untuk dicocokkan dan hanya berdasarkan pada memori. Diberikan titik query, akan ditemukan sejumlah k obyek atau (titik training) yang paling dekat dengan titik query. Klasifikasi menggunakan voting terbanyak diantara klasifikasi dari k obyek. Algoritma k-nearest neighbor (KNN) menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari query instance yang baru. Kita akan menerapkan KNN menggunakan data “Growth of Orange Trees”. Growth of Orange Trees adalah dataset yang menjelaskan tentang pertumbuhan pohon jeruk yang dilihat dari usia pohon serta lingkar batang pada pohon. Kerangka data Orange memiliki 35 baris dan 3 kolom catatan tentang pertumbuhan pohon jeruk.

library(class)  
Orange  
summary(Orange)

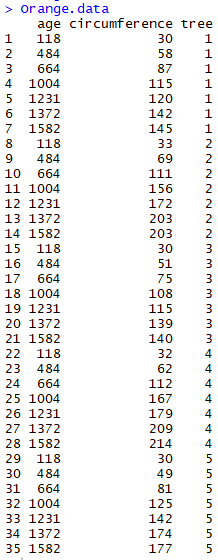
Pertama kita akan mengaktifkan *package library(class)*. Kemudian memanggil data *orange* yang sudah dijelaskan sebelumnya dan mencari informasi mengenai data tersebut menggunakan sintaks *summary(Orange)*. Berikut adalah *output*-nya:



*Tree*merupakan sebuah vaktor dengan level 1 sampai 5 yang menunjukan eksperimen yang bisa diterima oleh pohon jeruk dan terdapat 7 pohon jeruk pada masing-masing eksperimen. *Age*merupakan umur pohon jeruk dengan umur terendah yaitu 118 hari dan umur terpanjang yaitu 1582 hari. Sedangkan*Circumference*merupakan lingkar pada masing-masing batang pohon jeruk, dan lingkar pada pohon jeruk terkecil yaitu 30cm dengan lingkar batang pohon jeruk terbesar yaitu 214cm.

Selanjutnya saya akan membuat kelas data menggunakan sintaks berikut:

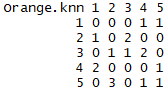
Orange.kelas<-c(rep("1",7),rep("2",7),rep("3",7),rep("4",7),rep("5",7))Orange.data<-data.frame(Orange[,2:3],tree=Orange.kelas)Orange.data



Gambar diatas merupakan kelas data pada *dataset “Growth of Orange Trees”*dengan menggunakan 2 variabel yaitu variabel “*Age*” dan variabel “*circumference*”.

Orange.knn<-knn(Orange.latihan[,-3],Orange.uji[,-3],Orange.latihan[,3],k=3)  
(table(Orange.knn,Orange.uji[,3]))

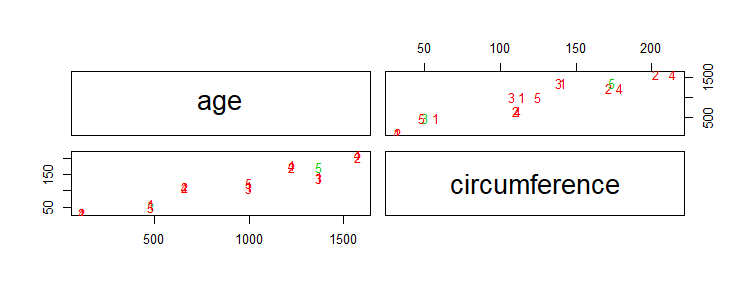
Sintaks diatas digunakan untuk menampilkan klasifikasi.



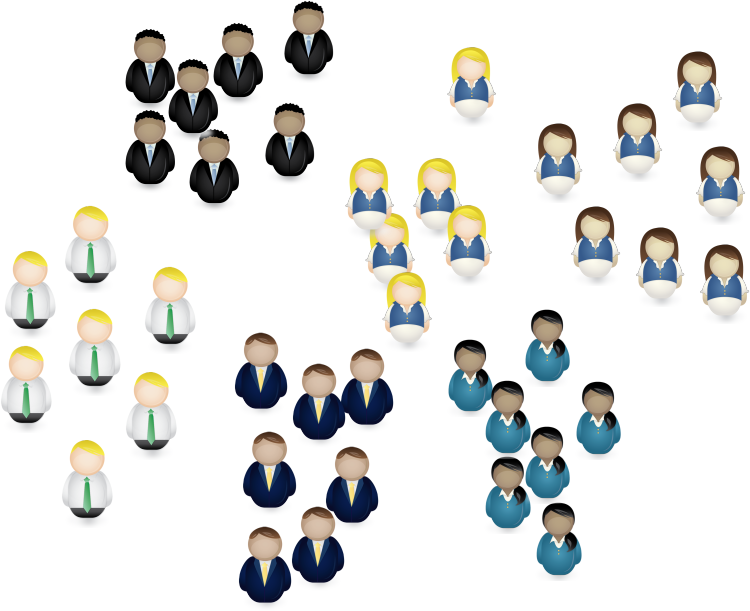
Dari *output* diatas dapat dilihat bahwa pohon pertama benar sebanyak 0 dengan salah sebanyak 1, pohon kedua benar sebanyak 0 dan salah sebanyak 3, pohon ketiga benar sebanyak 1 dan salah sebanyak 5, pohon 4 benar sebanyak 0 dan salah sebanyak 3, serta pohon lima benar sebanyak 1 dan salah sebanyak 3.

Kemudian kita akan *plot cluster*-nya.

pairs(Orange.uji[,1:2],pch=as.character(Orange.uji[,3]),col=c(3,2)  
 [(Orange.uji$tree!=Orange.knn)+1])

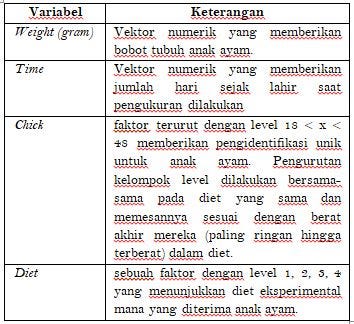
Dari *plot* diatas terlihat bahwa terdapat warna merah dari variabel “*Age*” serta “*circumference*” yang berwarna hijau hanya terdapat pada pohon 3 dan 5 dari setiap variabel “*Age*” maupun “*Circumference*”.

# Contoh Kasus : K-Nearest Neighbors (KNN) with R



K-Nearest Neighbors atau 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.

**Weight versus age of chicks on different diets**merupakan merupakan salah satu dataset yang terdapat dalam R. Weight versus age of chicks on different diets adalah kerangka data ChickWeight memiliki 578 baris dan 4 kolom dari percobaan tentang efek diet pada pertumbuhan awal anak ayam.

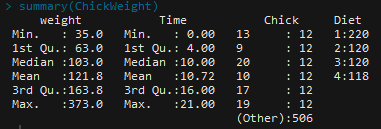


Dataset Weight versus age of chicks on different diets terdiri dari 578 kolom dan 4 baris. Untuk melakukan pengecekan jumlah baris dan kolom dapat dilakukan dengan perintah “dim(ChickWeight)”.

library(class)  
ChickWeight <-ChickWeight  
ChickWeight  
str(ChickWeight)  
dim(ChickWeight)

Untuk mengetahui deskriptif dari dataset Weight versus age of chicks on different diets dapat dilakukan dengan menggunakan perintah “summary(ChickWeight)”.

summary(ChickWeight)



Weight yang merupakan vector numerik yang memberikan bobot pada tubuh anak ayam dengan satuan gram memiliki bobot tubuh ayam terberat sebesar 373 grm, bobot tubuh anak ayam terkecil sebesar 35 grm, dan rata-rata bobot anak ayam sebesar 121.8 gram.

Time yang merupakan vaktor numerik dari jumlah hari saat ayam menetas saat dilakukan pengukuran memiliki jumlah hari telama sebesar 21 hari dengan rata-rata hari menetas sebesar 10.72.

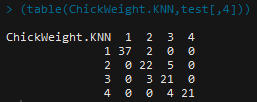
Diet yang merupakan sebuah faktor dengan level 1 sampai 4 yang menunjukkan diet eksperimen yang bisa di terima oleh anak ayam. Pada level pertama terdapat 220 anak ayam yang menerima, level kedua dan ketiga terdapat 120 anak ayam yang menerima, dan level keempat terdapat 118 anak ayam yang menerima.

Dalam pembuatan K-Nerest Neaighbors diperlukan pembagian data yaitu data train dan data test atau yang biasa dikenal dengan metode cross validation. Pada umumnya data train sebesar 80% dan data test sebesar 20%. Untuk membagi data dapat dilakukan dengan menggunakan perintah :

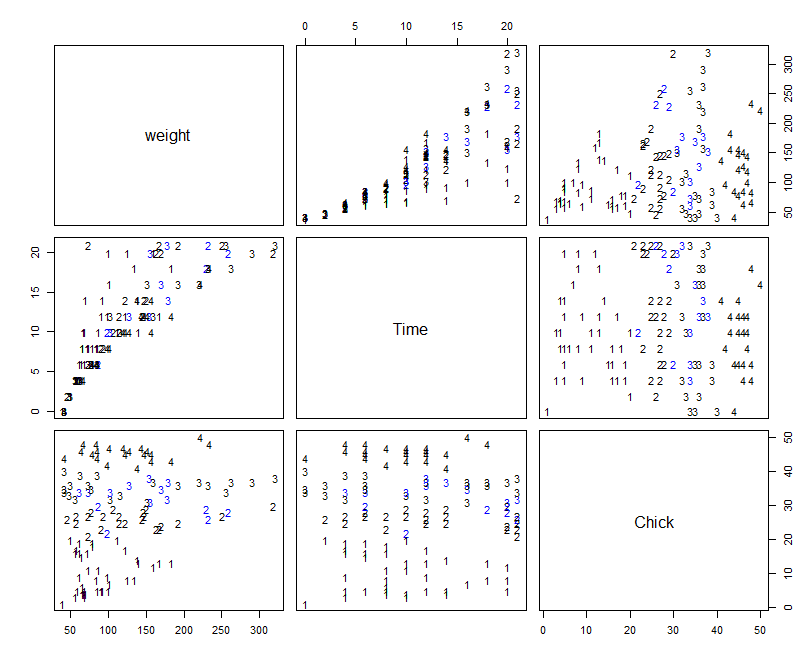
indexes = sample(1:nrow(ChickWeight), size = 0.2\*nrow(ChickWeight))  
#Test  
test = ChickWeight[indexes,]  
head(test)  
#Dimensi Test  
dim(test)  
#Train  
train = ChickWeight[-indexes,]  
head(train)

Selanjutnya, setelah data test dilakukan proses silang dimana data pengujian lantas dijadikan data train ataupun sebaliknya, data train sebelumnya dijadikan kini menjadi data test. Berikut sintaks yang digunakan:

#Dimensi Train  
dim(train)  
ChickWeight.KNN<-knn(train[,-4],test[,-4],train[,4],k=4)  
ChickWeight.KNN  
(table(ChickWeight.KNN,test[,4]))



Berdasarkan crosstab validation diatas dapat dilihat bahwa hasil diet level 1 benar sebesar 37 dan mengalami kesalahan sebesar 2, diet level 2 benar sebesar 22 dan mengalami kesalahan sebesar 5, diet level 3 benar sebesar 21 dan mengalami kesalahan sebesar 3, dan diet level 4 benar sebesar 21 dan mengalami kesalahan sebesar 4.



Untuk mendapatkan hasil visualisasi KNN seperti pada gambar 1.3 maka dapat menggunakan perintah :

pairs(test[,1:3],pch=as.character(test[,4]),col=c(1,4)[(test$Diet!=ChickWeight.KNN)+1])

Visualisasi seperti gambar diatas merupakan visualisasi pairs plot atau yang biasa dikenal dengan matrik scatterplot. Matriks scatterplot adalah cara yang bagus untuk menentukan secara kasar apakah terdapat korelasi linier antara beberapa variabel. Ini sangat membantu dalam menentukan dengan tepat variabel-variabel tertentu yang mungkin memiliki korelasi serupa dengan data genom atau proteomic.

Berdasarkan gambar 1.3 dapat dilihat bahwa terdapat cerimanan scatterplot pada garis diagonal. Jika dilihat bahwa terdapat korelasi linier pada variabel weight dan time karena dapat dilihat bahwa plot nya membentuk seperti garis. Sedangkan pada variabel weight dengan chick, dan time dengan chinck tidak terdapat korelasi linier karena terlihat penyebaran data (level diet).

1. **Latihan Praktikum**

Data yang digunakan dalam Praktikum kali ini adalah **Pima Indians Diabetes Database** Sumber: https://www.jair.org/index.php/jair/article/view/10129

**Langkah 1: Memanggil Package yang digunakan**

Pada praktikum ini akan digunakan ggplot2, caret, class, mvtnorm, MASS, dan gridExtra.

library(caret)

library(class)

library(ggplot2)

### **Langkah 2: Import & Explor Data**

*# alamat <- "ganti alamat dengan lokasi tempat data disimpan"*

alamat <- 'D:/pima-indians-diabetes.csv'

diabetes <- read.csv(alamat, stringsAsFactors = TRUE)

str(diabetes)

## 'data.frame': 768 obs. of 9 variables:

## $ preg : int 6 1 8 1 0 5 3 10 2 8 ...

## $ plas : int 148 85 183 89 137 116 78 115 197 125 ...

## $ pres : int 72 66 64 66 40 74 50 0 70 96 ...

## $ skin : int 35 29 0 23 35 0 32 0 45 0 ...

## $ insu : int 0 0 0 94 168 0 88 0 543 0 ...

## $ mass : num 33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...

## $ pedi : num 0.627 0.351 0.672 0.167 2.288 ...

## $ age : int 50 31 32 21 33 30 26 29 53 54 ...

## $ class: Factor w/ 2 levels "tested\_negative",..: 2 1 2 1 2 1 2 1 2 2 ...

diabetes$class <- as.factor(diabetes$class)

table(diabetes$class)

##

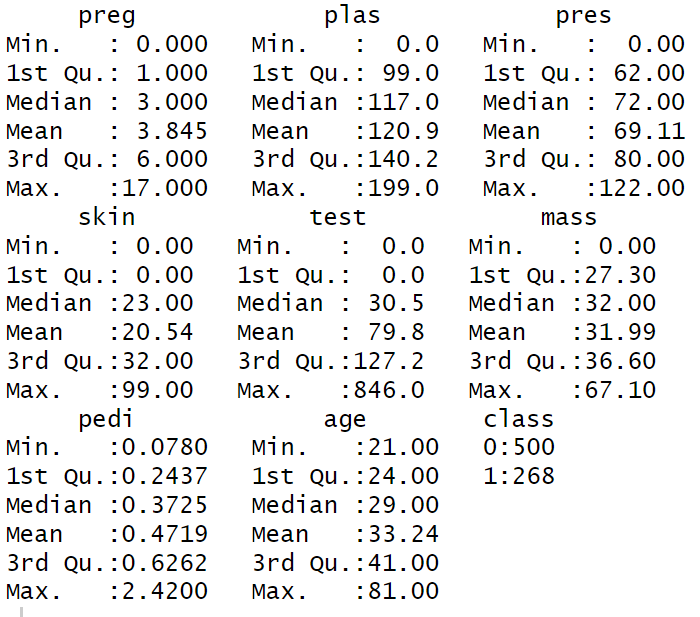
## tested\_negative tested\_positive

## 500 268

mean(diabetes$age)

## 33.24089

summary(diabetes$age)



### **Langkah 3: k-Nearest Neighbor (k-NN)**

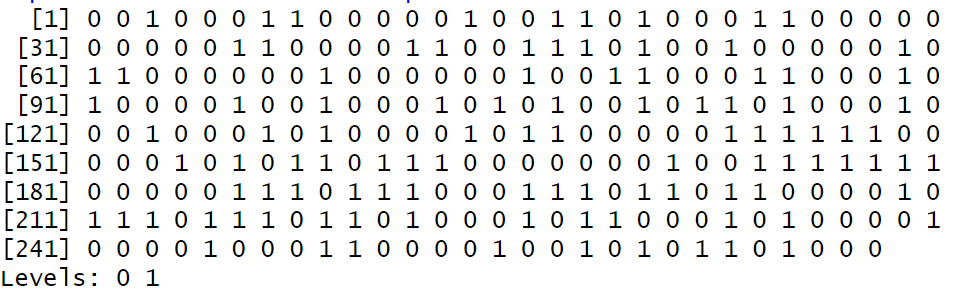
### k-NN melakukan klasifikasi berdasarkan k tetangga terdekat, sehingga sangat tergantung pada jarak. Sehingga, jika skala dan rentang peubah yang digunakan berbeda-beda, perlu melakukan standarisasi terhadap peubah tersebut. Umumnya Standarisasi yang paling sering digunakan adalah Standarisasi [0,1].

train=diabetes[1:500,]

test=diabetes[501:768,]

pred\_test=knn(train[,-9],test[,-9],train$class,k=2)

pred\_test

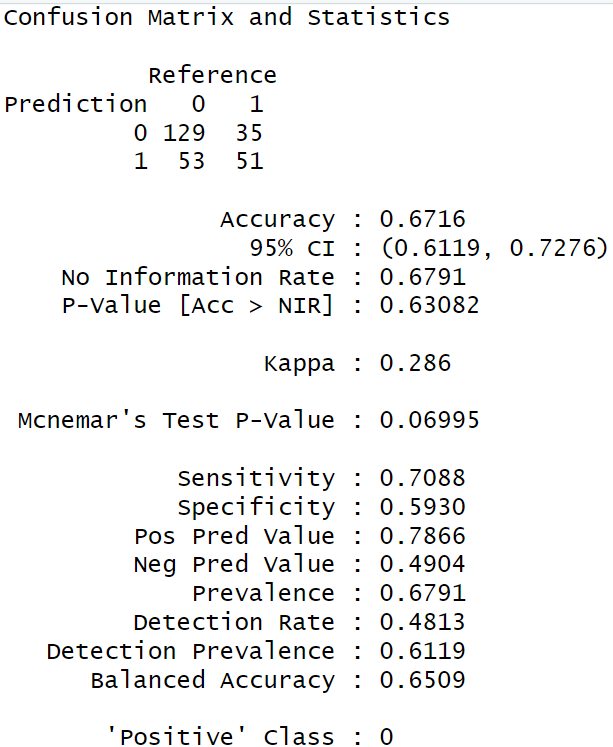
****

confusion=table(pred\_test,test$class)

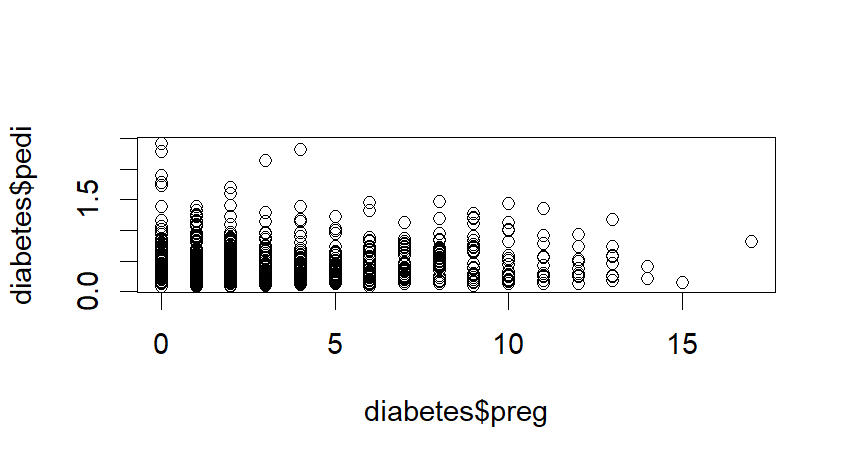
sum(diag(confusion))/nrow(test)

****

confusionMatrix(pred\_test,test$class)

****

plot(diabetes$preg,diabetes$pedi)

****

#standardize

stdmaxmin <- function(X) (X-min(X))/(max(X)-min(X))

preg1 <- stdmaxmin(diabetes$preg)

pedi1 <- stdmaxmin(diabetes$pedi)

m <-NULL; a <-b <-seq(0, 1, length.out = 70)

for (i in a) for (j in b) m <-rbind(m, c(i, j))

#k=3

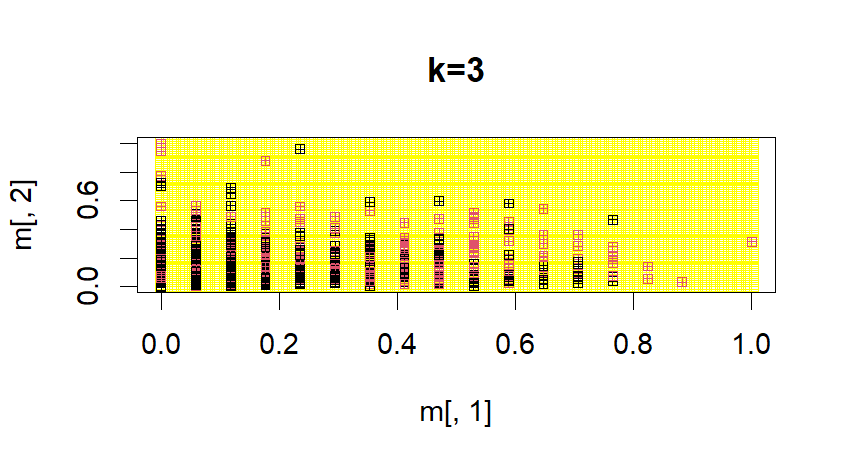
prediksi<-knn(cbind(preg1,pedi1), m, diabetes$class, k = 3)

plot(m[,1], m[,2], col=ifelse(prediksi=="tested\_positive", "cyan","yellow"),

pch=ifelse(prediksi=="tested\_positive",17,12), main="k=3")

points(preg1, pedi1, col=diabetes$class,

pch=ifelse(diabetes$class=="tested\_positive",17,12), cex=.7)

****

#k=7

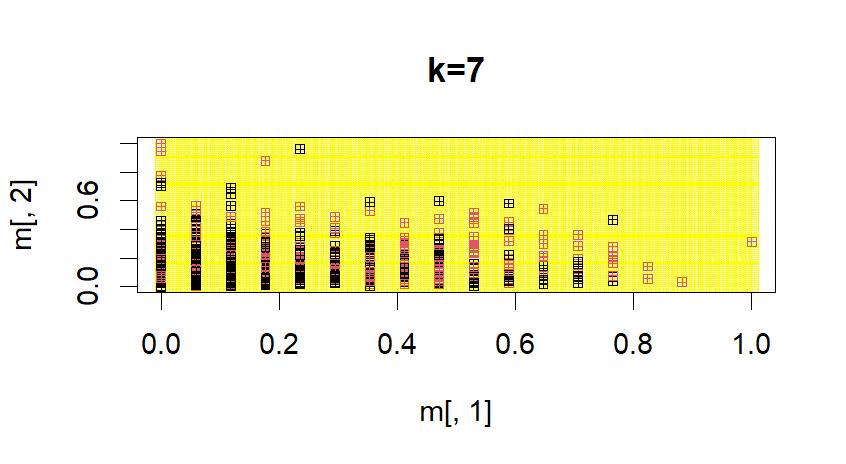
prediksi<-knn(cbind(preg1,pedi1), m, diabetes$class, k = 7)

plot(m[,1], m[,2], col=ifelse(prediksi=="tested\_positive", "cyan","yellow"),

pch=ifelse(prediksi=="tested\_positive",17,12), main="k=7")

points(preg1, pedi1, col=diabetes$class,

pch=ifelse(diabetes$class=="tested\_positive",17,12), cex=.7)

****

#k=10

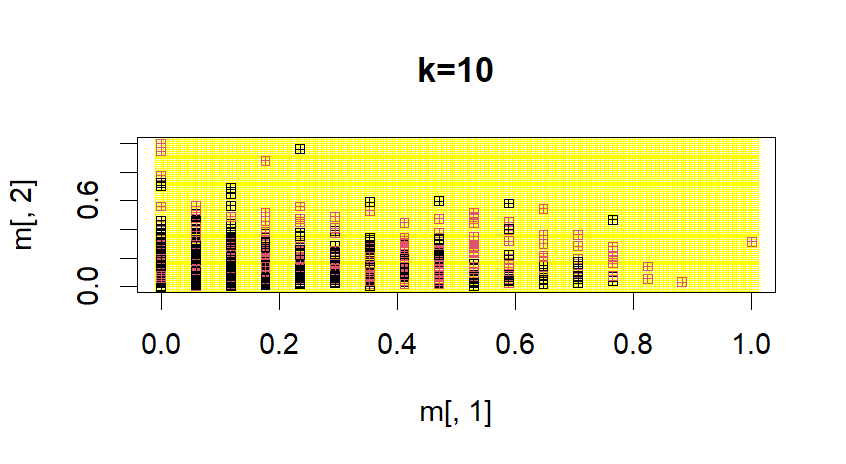
prediksi<-knn(cbind(preg1,pedi1), m, diabetes$class, k = 10)

plot(m[,1], m[,2], col=ifelse(prediksi=="tested\_positive", "cyan","yellow"),

pch=ifelse(prediksi=="tested\_positive",17,12), main="k=10")

points(preg1, pedi1, col=diabetes$class,

pch=ifelse(diabetes$class=="tested\_positive",17,12), cex=.7)

****

Dari hasil visualisasi dengan k yang berbeda-beda, terlihat hasilnya juga berbeda-beda. Semakin besar k, maka luasan wilayah akan semakin besar dan jarak batas umumnya lebih jauh dari data. Namun jika dicermati lebih jauh, semakin banyak juga data yang diklasifikasikan berbeda.