Machine Learning

Clasificacion supervisada usando los vecinos mas cercanos (KNN)

Dr. Edgar Acuna
Departmento de Matematicas

Universidad de Puerto Rico- Mayaguez

academic.uprm.edu/eacuna

En el método de los k vecinos mas cercanos (“knn: k nearest neighbors") (Fix y Hodges, 1951) se estima la función de densidad de donde provienen un conjunto de datos.

Aplicado a clasificacion supervisada el metodo k-nn se usa para estimar la función de densidad condicional $f(\mathbf{x}/C_j)$, de las predictoras $\mathbf{x}$ por cada clase C_j.

Es un método de clasificación noparamétrico, ya que no se hace ninguna suposición distribucional acerca de las variables predictoras.

Estimacion de densidad univariada por knn

- Sea x_1, x_2,… ..x_n una muestra con una función densidad desconocida f(x) que se desea estimar y sea t un número real. Recordar que la probabilidad de que x caiga en el intervalo (t-h,t+h), puede ser aproximada por 2hf(t), donde f es la función de densidad y h es una constante cercana a cero.
- Por otro lado dicha probabilidad tambien puede ser estimada por k/n, donde k es el número de observaciones que caen en el intervalo (t-h,t+h). En estimacion k-nn, k es prefijado y h se determina de acuerdo a el.
- Más formalmente, sea d(x,y)=|x-y| la distancia usual entre los puntos x e y de la linea recta.

Supongamos que hemos calculados todas las distancias $d(x_i,t)=|x_i-t|$ y que estas son ordenadas de menor a mayor, tal como

$d_1(t) \leq d_2(t) \leq \ldots \ldots . \leq d_n(t)$

- Entonces el estimador de la funcion de densidad f en el punto t segun sus k vecinos mas cercanos esta dado por

$$\hat{f}(t) = \frac{k}{2nd_k(t)}$$

Ejemplo

Estimar usando k-nn la función de densidad correspondiente al siguiente conjunto de datos.

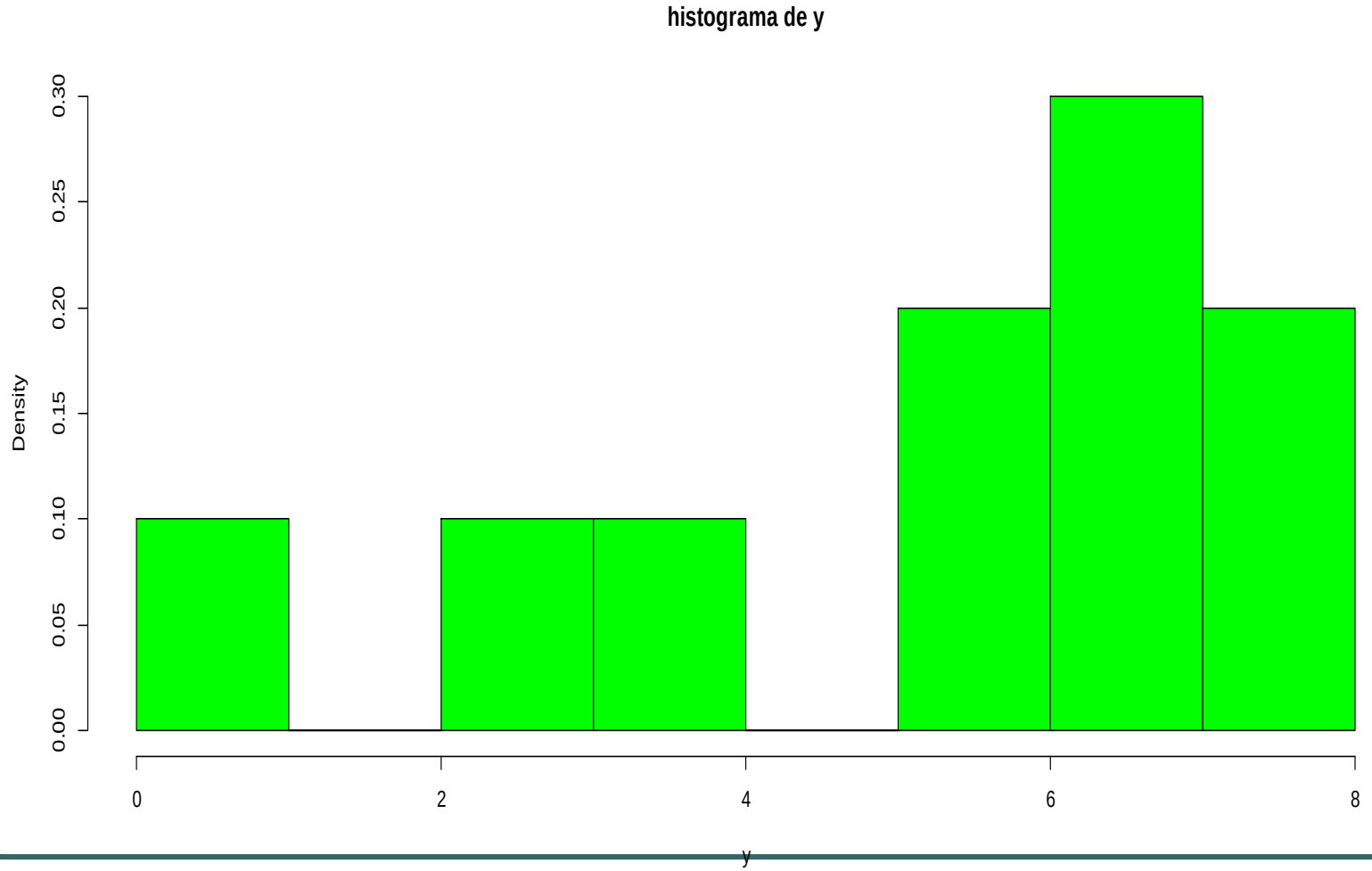
7.03860 6.38018 6.95461 2.25521 7.24608 6.62930 3.92734 0.40701 5.59448 5.05751

El histograma (normalizado para que el area sea 1) de los datos se muestra en el siguiente slide.

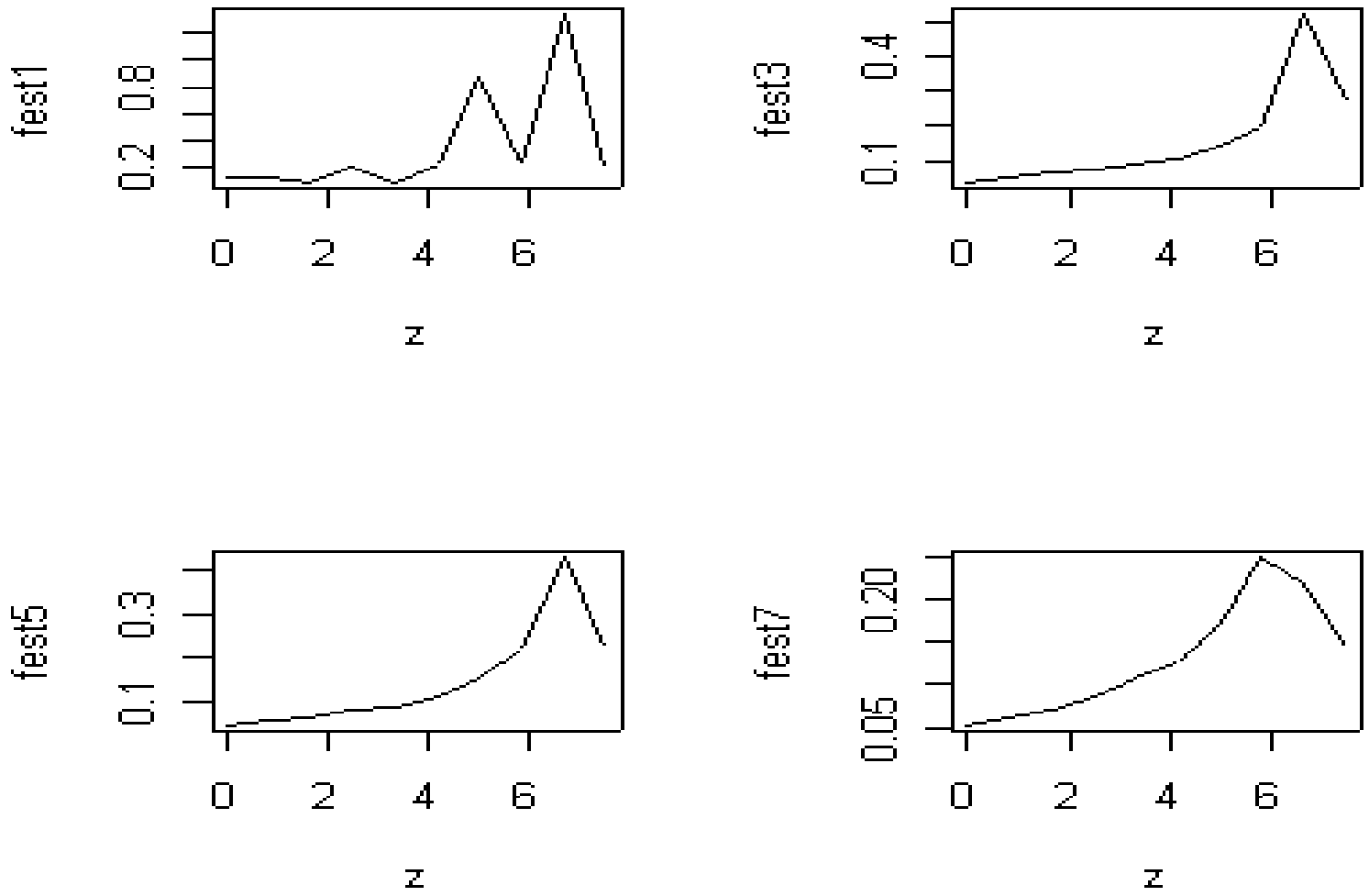
hist(y,freq=F,col="green")

El estimador de la funcion de densidad usando un vecino mas cercano se muestra abajo

```
>y=c(7.03860,6.38018,6.95461,2.25521,7.24608,6.62930,3.92734,0.40701,5.59448,5.05751)
> fdknn(y,10,1)
 [1] 0.05000000 2.79624682 0.05656350 0.50938975 0.08448023 0.12811826
 [7] 0.20642959 0.50845160 2.79624682 0.05000000
```


histograma de y
Density
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
2
4
6
8
y

Estimador k-nn de la funcion de densidad para k=1,3,5,7



Notar que cuando k es pequeno la curva sigue mucho a los puntos (Sobreajuste) y que cuando k es grande sigue mucho a la tendencia(sub-ajuste). Las dos cosas son malas y uno tienen que buscar un termino intermedio, porque sino las predicciones serian malas.

Cuando hay sobreajuste hay poco sesgo pero mucha variabilidad.
Cuando hay subajuste hay bastante sesgo pero poca variabilidad.

Cuando se estima el error por validacion cruzada se detecta si hay overfitting .

Estimacion de densidad multivariada

- El estimado de la función de densidad tiene la forma

$$\hat{f}(\mathrm{x}) = \frac{k}{n v_k(\mathrm{x})}$$

- donde $v_k(\mathbf{x})$ es el volumen de un elipsoide centrado en **x** de radio $r_k(\mathbf{x})$, que a su vez es la distancia de x al k-ésimo punto más cercano.

El clasificador k-nn

Desde el punto de vista de clasificacion supervisada el método k-nn es bien simple de aplicar (Cover y Hart, 1967)
En efecto, si las funciones de densidades condicionales $f(\mathbf{x}/C_i)$ de la clase C_i que aparecen en la ecuación

$$P(C_i / \mathrm{x}) = \frac{f(\mathrm{x} / C_i)\pi_i}{f(\mathrm{x})}$$

son estimadas por k-nn. Entonces, para clasificar un objeto, con mediciones dadas por el vector **x,** en la clase Ci se debe cumplir que

$$\frac{k_i \pi_i}{n_i v_k(\mathrm{x})} > \frac{k_j \pi_j}{n_j v_k(x)}$$

para $j \neq i$. Donde k_i y k_j son los k vecinos de x en las clase C_i y C_j respectivamente.

Asumiendo priors proporcionales a los tamaños de las clases (n_i/n y n_j/n respectivamente) lo anterior es equivalente a:

$$k_i > k_j \text{ para } j \neq i$$

Luego, el procedimiento de clasificación sería así:

1) Hallar los k objetos que están a una distancia más cercana al objeto **x**, k usualmente es un número impar 1 o 3.

2) Si la mayoría de esos k objetos pertenecen a la clase C_i entonces el objeto **x** es asignado a ella.

En caso de empate se clasifica al azar.

- Hay dos problemas en el método k-nn, la elección de la distancia o métrica y la elección de k.
- La métrica más elemental que se puede elegir es la euclideana d(**x**,**y**)=(**x**-**y**)'(**x**-**y**). Esta métrica sin embargo, puede causar problemas si las variables predictoras han sido medidas en unidades muy distintas entre sí. Algunos prefieren rescalar los datos antes de aplicar el método. Otra distancia bien usada es la distancia Manhattan definida por d(**x**,**y**)=|**x**-**y**|. Predictoras discretas pueden ser continuizadas.
- Enas y Choi (1996) usando simulación hicieron un estudio para determinar el k óptimo cuando solo hay dos clases presentes y determinaron que si los tamaños muestrales de las dos clases son comparables entonces $k=n^{3/8}$ si habia poca diferencia entre las matrices de covarianzas de los grupos y $k=n^{2/8}$ si habia bastante diferencia entre las matrices de covarianzas.

La distancia Gower (I)

- La distancia Gower (Gower, 1971) entre las instancias d-dimensionales x_i y x_j esta definida por

$$dist.Gow(x_i, x_j) = \frac{\sum_{k=1}^{d} w_{ijk} d_{ijk}}{\sum_{k=1}^{d} w_{ijk}}$$

- Si la k-esima variable es nominal entonces d_{ijk}=0 si x_{ik}=x_{jk} y 1 en caso contrario . Los $w_{ijk=}$1
- Si la k-esima variable es logica o binaria asimetrica entonces d_{ijk}=0 si x_{ik}=x_{jk} =TRUE, y 1 en caso contrario . Los w_{ijk} seran 1 en todos los casos excepto cuando x_{ik}=x_{jk} =FALSE en cuyo caso valen 0.
- Si la k-esima variable es continua u ordinal entonces $d_{ijk=}$|x_{ik}-x_{jk}|/rango(x_k). Los w_{ijk}=1. Tambien si x_{ik} o x_{ik} o ambos estan "missing" entonces w_{ijk} vale 0.

- El sesgo del error de classificación aumenta a medida que k aumenta, en tanto que la varianza disminuye (subajuste).
- La tasa de error tiende a aumentar cuando se elige un mayor numero de vecinos.
- Se ha demostrado (Cover y Hart, 1967) que la tasa de error del clasificador k-nn es a lo más dos veces la tasa de error óptimo (error del clasificador Bayesiano donde las posteriores son conocidas).

La Distancia Gower

La funcion gower.dist del paquete StatMatch de R calcula la distancia gower

Gower.dist(ejearbol)

Otra manera de lidiar con variables nominales es usar variables binarias para representarlas. Asi si una variable nominal asume k valores distintos se pueden usar k-1 variables binarias para representarlas.

Wilson y Martinz (1997 y 2000) tambien han propuesto tres otras medidas: HVDM, IVDM,WVDM

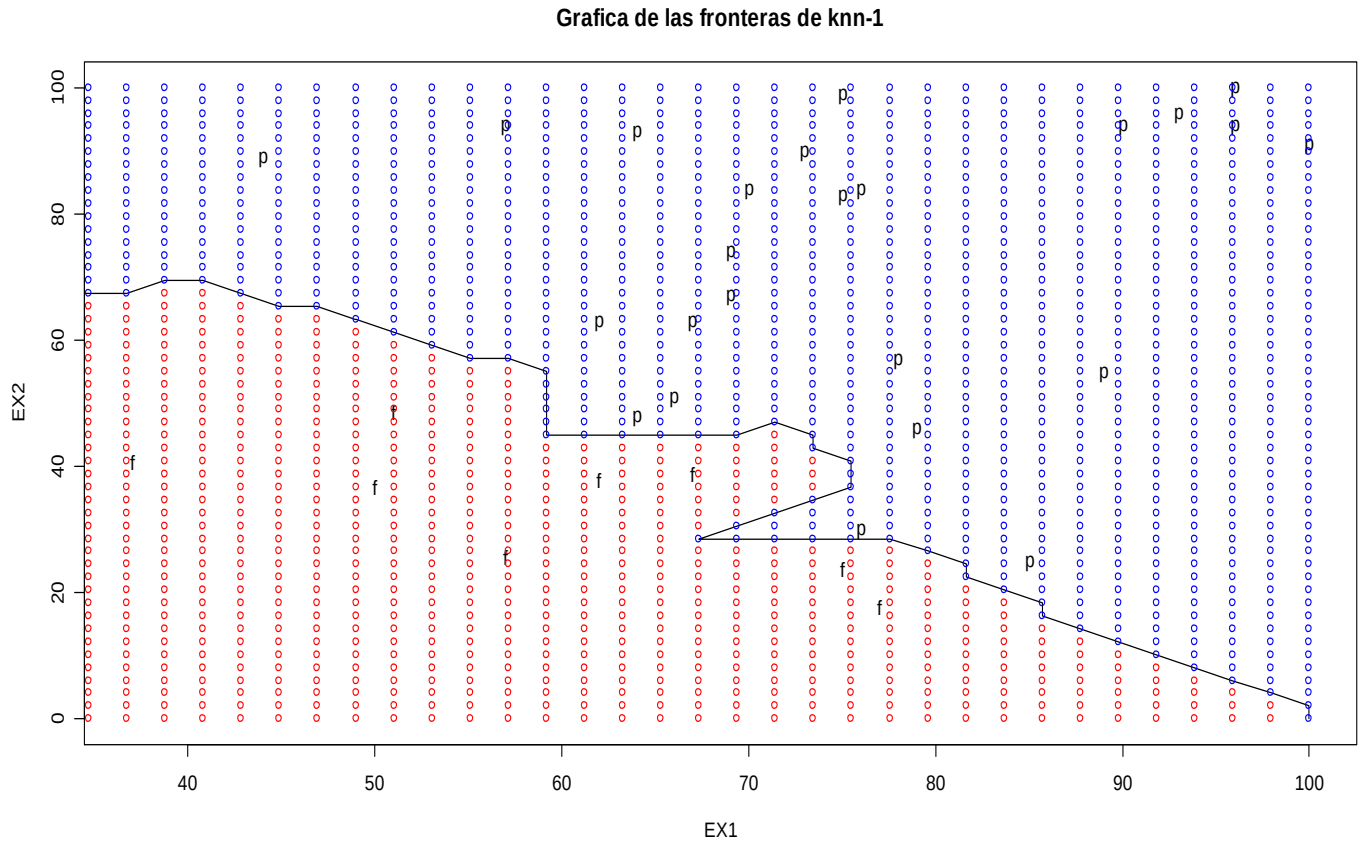
Wojna (2005), modifico la distancia WVDM y la llamo DWVDM

Ejemplo: Datos de notas

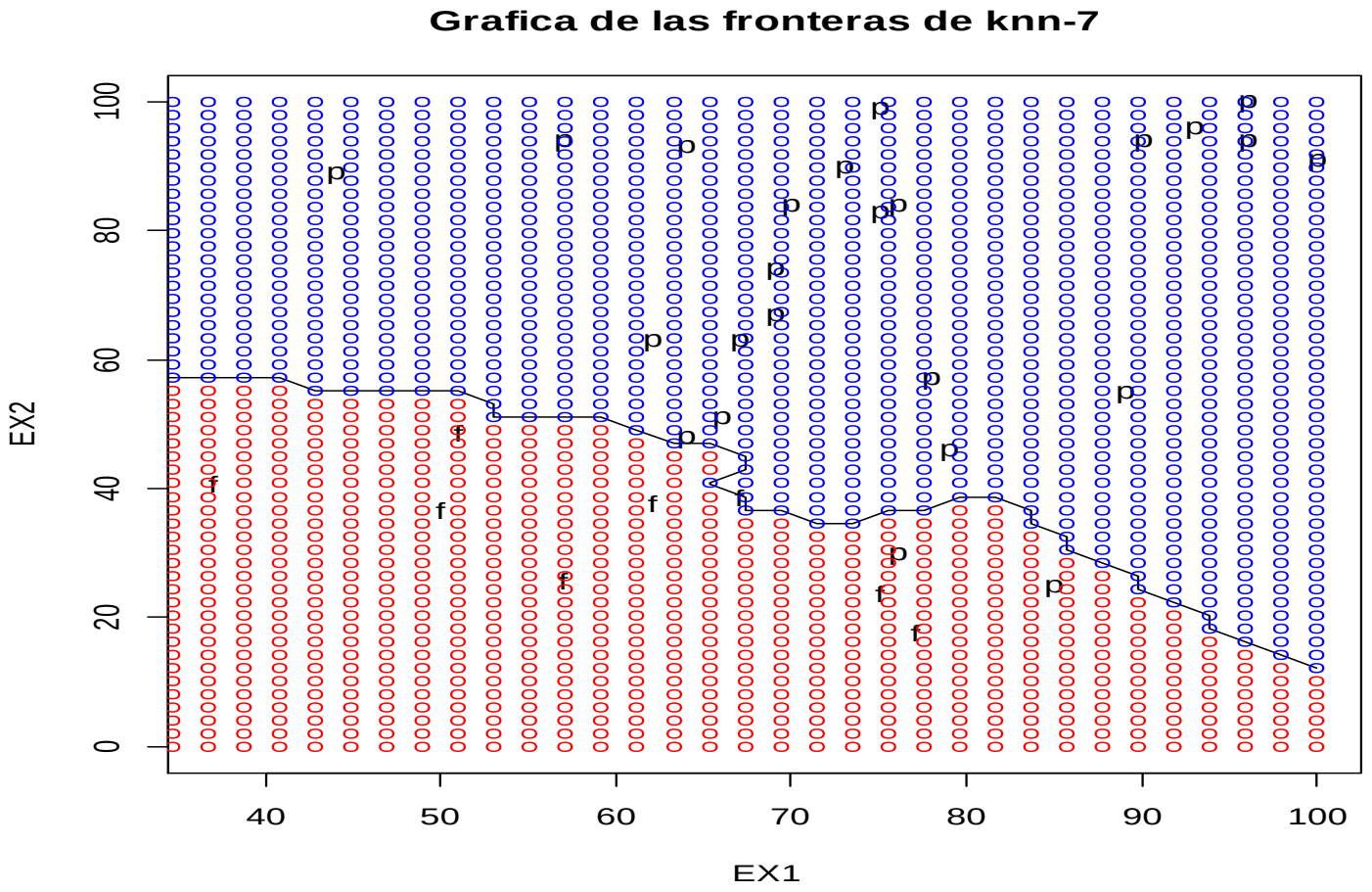
eje1dis=read.table("http://math.uprm.edu/~edgar/eje1disc.dat",header=T)

```
> knn(eje1dis[,1:2],eje1dis[,1:2],eje1dis[,3],k=1)
 [1] p p p p p p p p p p p p p p p p p p p p p p p p p f f f f f f f f
Levels: f p
>#  errror de clasificacion con k=1 vecinos
> exak1=knn(eje1dis[,1:2],eje1dis[,1:3],eje1dis[,4],k=1)
> mean(exak1!=eje1dis[,4])
[1] 0
> knn(eje1dis[,1:2],eje1dis[,1:2],eje1dis[,3],k=5)
 [1] p p p p p p p p p p p p p p p p f p p p p f p p p f f p f f f f f
Levels: f p
>#  error de clasificacion  con k=5 vecinos
>exak5=knn(eje1dis[,1:2],eje1dis[,1:2],eje1dis[,3],k=5)
> mean(exak5!=eje1dis[,4])
[1] 0.09375
```

Ejemplo: Frontera de decision



Ejemplo: Frontera de decision



Ejemplo de knn: Bupa

```
> knn(bupa[,1:6],bupa[,1:6],bupa[,7],k=3)
  [1] 2 2 2 2 2 2 1 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 1 1 2 1 1 1 2 1 1 1 2 1 2 1
 [38] 2 2 2 2 2 1 2 1 2 2 2 1 2 1 1 2 2 2 2 2 1 2 1 2 2 2 2 1 1 2 2 2 2 2 1 1 1
 [75] 1 1 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1
[112] 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 1 2 1 1 2 1 2 2
[149] 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 1 2 1 1 2 1 2 2 1 2 2 1 1 2 2
[186] 2 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 1 2 1 2 2 2 2 2
[223] 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 2 2 2 1 2 2 1 2 1 1
[260] 2 1 2 1 2 2 2 2 2 1 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2
[297] 2 1 1 2 1 2 2 2 2 2 2 1 2 2 2 1 2 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 2 2 2 1 2
[334] 2 1 2 2 2 1 2 2 1 1 1 2
Levels: 1 2
Error de mala clasification por resustitucion
> mean(knn(bupa[,1:6],bupa[,1:6],bupa[,7],k=3)!=bupa[,7])
[1] 0.1971014
```

Ejemplo (cont)

```
> bupak1=knn(bupa[,-7],bupa[,-7],as.factor(bupa[,7]),k=1)
> table(bupak1,bupa[,7])
bupak1   1   2
     1   145   0
     2     0 200
error=0%
> bupak3=knn(bupa[,-7],bupa[,-7],as.factor(bupa[,7]),k=3)
> table(bupak3,bupa[,7])
      bupak3   1   2
     1       106  29
     2        39 171
error=19.71%
> bupak5=knn(bupa[,-7],bupa[,-7],as.factor(bupa[,7]),k=5)
> table(bupak5,bupa[,7])
bupak5   1   2
     1  94  23
     2  51 177
error=21.44%
```

Ejemplo (cont.)

```
> mean(knn.cv(bupa[,1:6],bupa[,7],k=5)!=bupa[,7])
[1] 0.3391304
> mean(knn.cv(bupa[,1:6],bupa[,7],k=7)!=bupa[,7])
[1] 0.3130435
> mean(knn.cv(bupa[,1:6],bupa[,7],k=9)!=bupa[,7])
[1] 0.3043478
> mean(knn.cv(bupa[,1:6],bupa[,7],k=11)!=bupa[,7])
[1] 0.3188406
> crossval(bupa,method="knn",kvec=7,repet=10)
[1] 0.3133333
> crossval(bupa,method="knn",kvec=9,repet=10)
[1] 0.3084058
> crossval(bupa,method="knn",kvec=11,repet=10)
[1] 0.3226087
```

Ejemplo de knn: diabetes

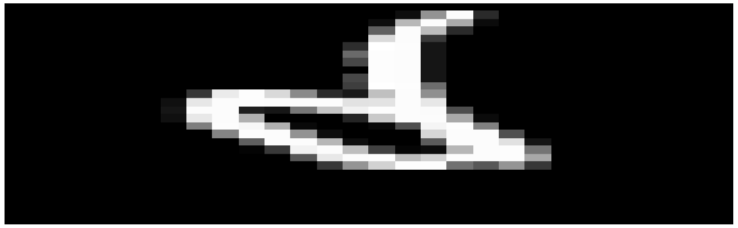
```
> diabk3=knn(diabetes[,-9],diabetes[,-9],as.factor(diabetes[,9]),k=3)
> table(diabk3,diabetes[,9])

diabk1   1   2
     1 459  67
     2  41 201
error=14.06%
> diabk5=knn(diabetes[,-9],diabetes[,-9],as.factor(diabetes[,9]),k=5)
> table(diabk5,diabetes[,9])

diabk1   1   2
     1 442  93
     2  58 175
error=19.66
```

Aplicacion de knn a reconocimiento de digitos

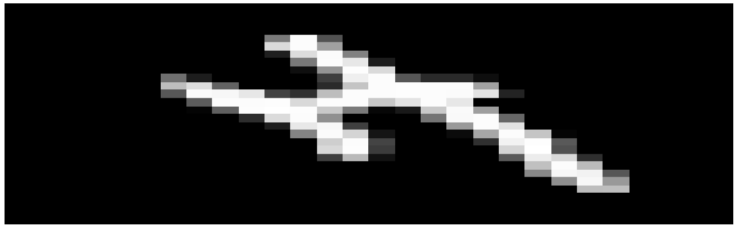
The MNIST dataset de LeCun, Cortes y Burges, Consiste de 60 mil imagenes de digitos del 0 al 9 en la muestra de entrenamiento y 10 mil en la muestra de prueba. Cada imagen conside 764 pixels (28x28) y la primera de las columnas es la clase.
Los datos estan en formato csv en https://pjreddie.com/ The training es 104MB y el test es 17MB
Visualizando los digitos: La siguiente function muestar la imagen del digito que aparece en la fila obs de la matriz de pixels mat

```
imagePlot=function (mat,obs) {
    m <- matrix(as.numeric(obs[-1]), 28, 28)
    image(m, axes = FALSE, col = grey(seq(0, 1, length = 256)))
    title(main = mat[obs,1])
}
```

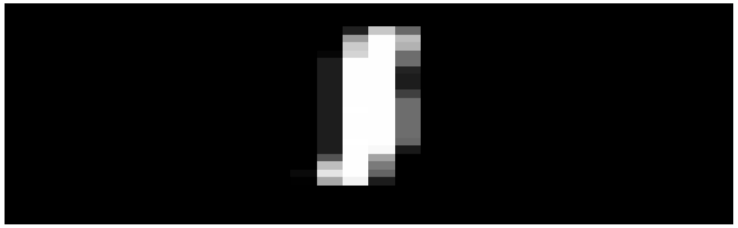
9



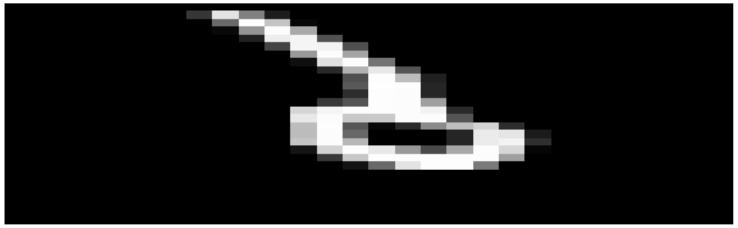
4



1



9



Knn para un subset

```
#usaremos una muestra de 20 mil imagenes del training original
#15,000 instancias como training y 5,000 como test.
a=fread("http://www.pjreddie.com/media/files/mnist_train.csv")
small=a[1:20000,]
sample=sample(1:20000,15000)
train=small[sample,]
test=small[-sample,]
train=as.matrix(train)
test=as.matrix(test)
knn1=knn(train[,-1],test[,-1],train[,1])
mean(knn1!=test[,1])
[1] 0.0406
```

Reduciendo el tiempo de corrida de k-nn

- Complejidad O(ndk), donde n es el numero de instancias, d el numero de atributos predictores y k el numero de vecinos.
- Uso de kd-trees(1975). Baja la complejidad a (kdlog(n)). Se deteriora cuando la dimensionalidad aumenta. (Paquete RANN)
- Uso de Local Sensitivity Hashing (LSH) (2005). que es un algoritmo aleatorio que encuentra los vecinos mas cercanos con alta probabilidad.

Otro clasificador noparametrico

- Estimacion de densidad por kernel. Pero su mayor desventaja es que es pesado computacionalmente. El mas usado es el kernel gaussiano.
- Los metodos noparametricos por lo general afrontan el problema de la maldicion de la dimensionalidad. El numero de datos debe crecer casi exponencialmente al numero de la dimension para que la estimacion sea buena.