Reconocimiento de Patrones
Grado en Ingeniería Informática
maria.guerrero@uca.es
Departamento Ingeniería Informática

Simulacro de examen

PROBLEMA 1. En un hospital se analizó a 40 pacientes que presentaban un lunar sospechoso en alguna parte de su cuerpo, y se midió la irregularidad del borde del lunar, observándose su evolución posterior, para tratar de realizar un sistema de diagnóstico muy valioso para el hospital.

Se dispone de los datos de 200 pacientes, de los que 20 evolucionaron a cáncer maligno. Para simular estos datos, vamos a generar 200 pares <x,y> tal y como se indica:

```
randn('seed',0); rand('seed',0);
x=[2+randn(1,180) 4+0.2*randn(1,20)];
y=[zeros(1,180) ones(1,20)];
[x,y]=shuffle(x,y);
```

La tasa de aciertos en el proceso de diagnóstico es vital, ya que:

- Cada vez que un lunar es diagnosticado como "maligno", se aplica un tratamiento al paciente, cuyo coste es de 6.000 euros.
- Cada vez que un paciente es diagnosticado como sano, y finalmente el lunar era maligno por un error en el diagnóstico, el hospital debe pagar una indemnización de 100.000 euros a la familia.
- El coste de que el médico realice un diagnóstico es de 300 euros.

Suponiendo que las distribuciones de enfermos y de sanos son gaussianas (que de hecho lo son), se pide:

a. Definir la matriz de coste del proceso de decisión. Explique cada uno de los valores asignados a esta matriz de 2x2.
b. Explique cómo modificaría la matriz, si se supone que el coste del diagnóstico es despreciable (o sea, si se considera un coste para el diagnóstico igual a 0). Utilizaremos esta nueva matriz para el apartado c.
c. Determinar numéricamente la frontera de decisión óptima para la medida de la irregularidad del borde de un lunar, que permita clasificar como sanos a aquellos pacientes cuya irregularidad sea menor que la frontera, y como enfermos si la medida es mayor. En este apartado se considerará que únicamente disponemos de los datos <x,y>, y que los valores usados para generarlos se desconocen.

PROBLEMA 2. Para este problema trabajaremos con la base de datos denominada Housing Database (Boston). En dicha base de datos, cada dato (506 en total) ocupa una fila completa, y cada columna representa una característica (14 características).

Del total de 14 atributos, se desea predecir el último (MEDV) utilizando los 13 primeros. Realizar las siguientes operaciones previas:

1. Almacenar los datos en x e y, según el formato clásico. De tal forma, x deberá tener dimensiones 13x506 e y será de 1x506
2. Realizar una mezcla de los datos con la semilla dada:

```
rand('seed',1);
[xd,yd] = shuffle(x,y);
```

3. Utilizar las 400 primeras instancias de [xd,yd] como datos de entrenamiento(TRN) y las 106 últimas instancias, como datos de test (TST)

Se pide diseñar un **modelo de regresión lineal** para predecir el atributo MEDV a partir de los demás, y responder a las siguientes cuestiones:

a) ¿Cuál es el error de resustitución usando los datos TRN?
b) ¿Cuál es la estimación del error de generalización usando 10-CV sobre los datos TRN?
c) ¿Cuál es el error del modelo diseñado con los datos TRN si se prueba sobre los datos TST?

SOLUCIONES

% PROBLEMA 1

```
clc,clear all,close all
% APARTADO A)
%   Matriz de coste
%                                  Real
%                        |  Sano     | Enfermo |
%              ----------+-----------+---------+
%               Sano     |     A     |    C    |
%  Prediccion  ----------+-----------+---------+
%               Enfermo  |     B     |    D    |
%              ----------+-----------+---------+
A =    300;    % Diagnostico
B =   6300;    % Diagnostico + Tratamiento
C = 100300;    % Diagnostico + Indemnizacion
D =   6300;    % Diagnostico + Tratamiento

M = [A C;
     B D]

% APARTADO B)
M=M-300

% APARTADO C)
% Generar los datos
randn('seed',0); rand('seed',0);
x=[2+randn(1,180) 4+0.2*randn(1,20)];
y=[zeros(1,180) ones(1,20)];
[x,y]=shuffle(x,y);

% Calcular la media y desv. standard de cada distribucion
ind0 = find(y==0);   ind1 = find(y==1);
m0 = mean(x(ind0));  m1 = mean(x(ind1));
s0 = std(x(ind0));   s1 = std(x(ind1));

% Obtener la frontera
Pw0=length(ind0)/length(y);     Pw1=length(ind1)/length(y);
C0 = M(2,1)/sum(M(:));    C1 = M(1,2)/sum(M(:));
A=s0*s0-s1*s1;
B=2*(m0*s1*s1-m1*s0*s0);
C=2*s0*s0*s1*s1*(log(C0)-log(C1)+log(Pw0)-log(Pw1)-
log(s0)+log(s1))+s0*s0*m1*m1-s1*s1*m0*m0;
x1=(-B+sqrt(B*B-4*A*C))/2/A
x2=(-B-sqrt(B*B-4*A*C))/2/A

% La solución correcta es x2=3.4875, ya que usar x1=4.6714 implicaría
% que si el lunar es sumamente irregular(mayor que 4.67), se dé al
%paciente como sano. Esto se puede comprobar haciendo la
%representacion grafica de ambas distribuciones, teniendo en cuenta
%las prob. iniciales y el coste de las decisiones incorrectas
xi = -5:0.001:5;
D0 = normpdf(xi,m0,s0) * Pw0 * C0;
D1 = normpdf(xi,m1,s1) * Pw1 * C1;

plot(xi,D0,'r');hold on;plot(xi,D1,'b');hold off;
title(['Frontera = ' num2str(x2)])
```

```
% PROBLEMA 2
clc,clear all,close all

% Se crea el fichero de texto housing.data.txt
A=load('housing.data.txt');
x = A(:,1:13)';
y = A(:,14)';

rand('seed',1);
[xd,yd]=shuffle(x,y);
xtrn = xd(:,1:400);
xtst = xd(:,401:end);
ytrn = yd(1:400);
ytst = yd(401:end);

% Error de resustitucion. Observar que la salida se calcula con la
% misma M con la que se han calculado los coeficientes del modelo
% ================================================================
M1 = [xtrn' ones(size(xtrn,2),1)];  % Calculamos los coefs
coefs = pinv(M1) * ytrn';           % del modelo
M2=M1;
salida = M2 * coefs;                % Calculamos la salida
Err_resust = sumsqr(salida - ytrn') / length(ytrn)  % Calculamos el
error

% Error 10-CV usando solo los datos TRN
% =====================================
Err_CV = 0;
for i=1:10,
  [x1,x2,y1,y2]=crossval(xtrn,ytrn,10,i);
  M1 = [x1' ones(size(x1,2),1)];    % Calculamos los coefs. del modelo
  coefsCV = pinv(M1) * y1';
  M2 = [x2' ones(size(x2,2),1)];            % Calculamos la salida
  salida =  M2 * coefsCV;
  Err_CV = Err_CV + sumsqr(salida - y2');   % Acumulamos el error
end
Err_CV = Err_CV / length(ytrn)

% Error sobre datos TST
% =====================
M1 = [xtrn' ones(size(xtrn,2),1)];   % Calculamos los coefs. con los
datos TRN
coefs = pinv(M1) * ytrn';
M2 = [xtst' ones(size(xtst,2),1)];  % Calculamos la salida con los
datos de TST
salida =  M2 * coefs;
Err_TST = sumsqr(salida-ytst') / length(ytst)     % Obtenemos el error
```