5º INGENIERÍA DE TELECOMUNICACIÓN

# INTELIGENCIA ARTIFICIAL Y RECONOCIMIENTO DE PATRONES

## Práctica 10:

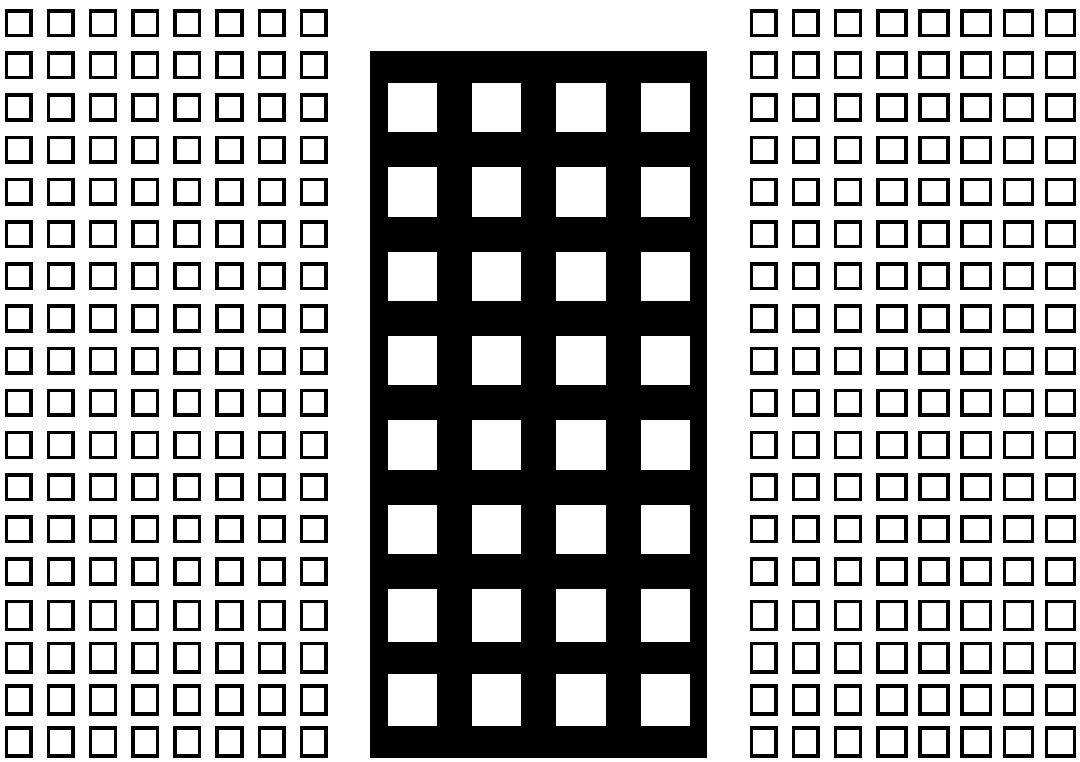
# Desarrollo de un control de acceso. Parte I: entrenamiento.

---

***Objetivos:***

- Comprender el funcionamiento de un sistema de control de acceso.
- Desarrollar la etapa de entrenamiento de un control de acceso.
- Elegir el número de componentes (PCA) y el clasificador para lograr un funcionamiento óptimo.

---



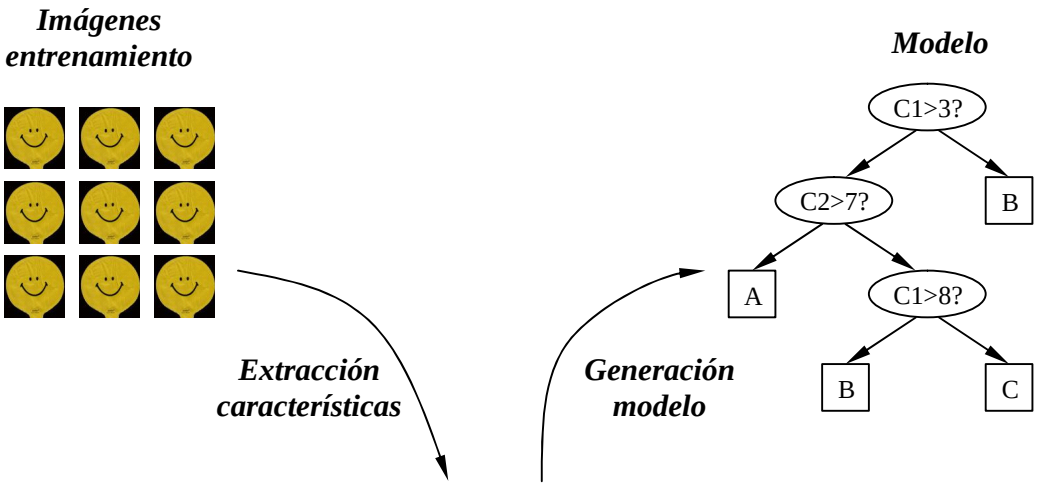
# 1. FUNCIONAMIENTO DE UN CONTROL DE ACCESO BASADO EN RECONOCIMIENTO DE CARAS.

Un control de acceso se estructura en dos fases:

**Fase 1: entrenamiento.**

En esta fase se recopilan imágenes de diversos sujetos (varias imágenes por sujeto con pequeñas variaciones); se extraen características a partir de ellas (proyecciones sobre el espacio formado por las componentes principales o PCA) y se genera un modelo que se ajuste a los ejemplos de entrenamiento (existen muchos modelos distintos: vecino más cercano, árbol de decisión, listas de reglas, métodos bayesianos, redes neuronales, etc.).

Tanto la recopilación de imágenes como la extracción de características ya se han realizado en prácticas anteriores. Se partirá de tales resultados.

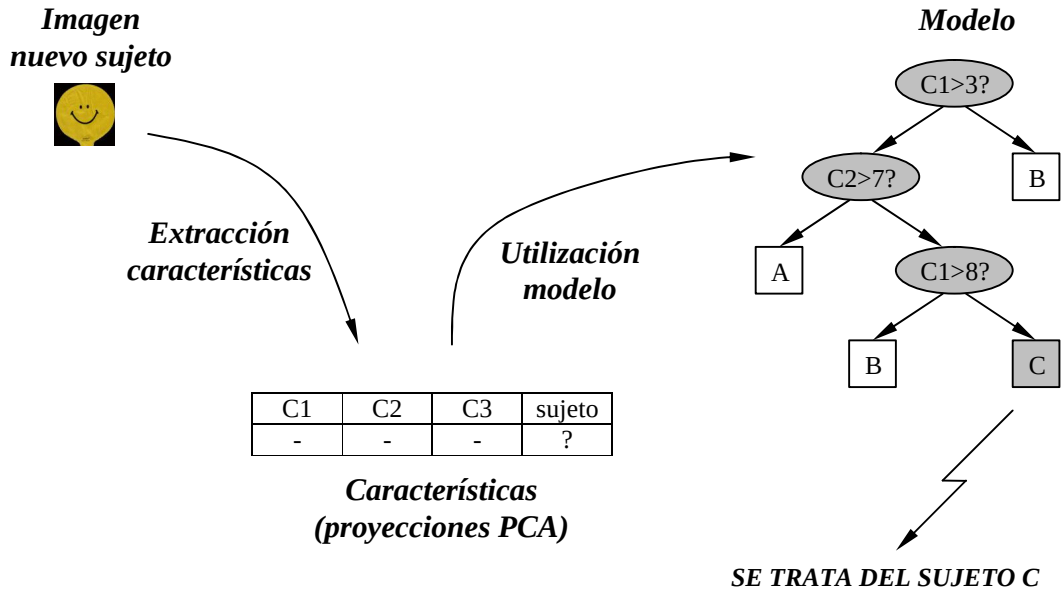



| C1 | C2 | C3 | sujeto |
|---|---|---|---|
| - | - | - | A |
| - | - | - | B |
| - | - | - | C |
| - | - | - | B |
| - | - | - | A |
| - | - | - | C |
| - | - | - | A |

***Características (proyecciones PCA)***

**Fase 2: reconocimiento.**

En esta fase se capta la imagen de un sujeto que supuestamente desea acceder a un edificio, y se intenta determinar su identidad. Para ello, primero se extraen la características de la imagen (proyectándola sobre el mismo espacio generado con las imágenes de entrenamiento) y se aplica el modelo también generado anteriormente para determinar qué clasificación se obtiene (a qué sujeto se parece más la imagen).




Esta práctica se centra en la fase de entrenamiento; la próxima práctica se centrará en la fase de reconocimiento.

# 2. CAPTURA DE IMÁGENES DE ENTRENAMIENTO Y EXTRACCIÓN DE CARACTERÍSTICAS.

Este proceso ya realizó en las prácticas 2 y 3. Se dispone de una base de datos de imágenes de 12 sujetos, de las características extraídas a partir de ellas y del espacio sobre el que se proyectaron las imágenes para extraer las características (este último dato se utilizará para el reconocimiento en la práctica siguiente).

Se trabajará sobre tres escenarios distintos, en orden creciente de dificultad:

- Escenario 1: contiene las fotos tomadas sobre fondo blanco.
- Escenario 1+2: contiene las fotos anteriores y además las fotos tomadas sobre fondo no uniforme.
- Escenario 1+2+3+4: contiene las fotos anteriores y además las fotos tomadas con oclusiones (gorras, gafas de sol, etc.) y todo tipo de fondos.

En cada caso, se puede trabajar con un número de componentes (número de características extraídas para cada imagen) variable entre 2 y 9.

Se proporcionan los siguientes ficheros de datos de Matlab:

| Escenario | Número de componentes extraídas | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | A2.mat | A3.mat | A4.mat | A5.mat | A6.mat | A7.mat | A8.mat | A9.mat |
| 1+2 | B2.mat | B3.mat | B4.mat | B5.mat | B6.mat | B7.mat | B8.mat | B9.mat |
| 1+2+3+4 | C2.mat | C3.mat | C4.mat | C5.mat | C6.mat | C7.mat | C8.mat | C9.mat |

Los ficheros se irán descargando a medida que sean necesarios; debe tenerse en cuenta que al cargar los datos de un fichero se sobrescriben los datos cargados con anterioridad. Todos ellos se encuentran comprimidos en un fichero zip y están accesibles desde la página web de la asignatura:

http://lorca.umh.es/isa/es/asignaturas/iarp/practicas/P10/caras.zip

# 3. GENERACIÓN AUTOMÁTICA DEL FICHERO DE DATOS PARA WEKA.

Para crear un modelo con la herramienta WEKA, los datos disponibles en variables de matlab han de escribirse como un fichero en formato .arff.

Hasta ahora los ficheros .arff han sido generados manualmente; pero esto es imposible cuando se trata de muchos datos o se desea un funcionamiento automático del sistema. A continuación se indica cómo mediante un programa de Matlab es posible escribir automáticamente un fichero .arff.

```
% escribe un fichero de datos arff
function escribe_arff(fichero_arff, fichero_matlab)

% carga datos
load(fichero_matlab);

% numero de datos
n_sujetos = 12;
n_carac = size(p1,1);
n_imag = size(p1,2);

% abre fichero escritura
file = fopen(fichero_arff, 'w');

% escribe la cabecera
fprintf(file, '@relation caras\n\n');
for i=1:n_carac
    fprintf(file, '@attribute carac_%02d numeric\n', i);
end
fprintf(file, '@attribute sujeto {');
```

```
for i=1:n_sujetos-1
    fprintf(file, '%02d, ', i);
end
fprintf(file, '%02d}\n\n', n_sujetos);
fprintf(file, '@data\n\n');

% escribe los datos
for i=1:n_sujetos
    orden = sprintf('p = p%d;', i);
    eval(orden);
    for j=1:n_imag
        for k=1:n_carac
            fprintf(file, '%f   ', p(k,j));
        end
        fprintf(file, '%02d \n', i);
    end
end

% cierra fichero
fclose(file);
```

El programa anterior también se facilita en la página web de la asignatura:

http://lorca.umh.es/isa/es/asignaturas/iarp/practicas/P10/escribe_arff.m

Con los ficheros en formato .arff es posible estudiar el funcionamiento de diversos clasificadores. A continuación se muestra un ejemplo de evaluación del funcionamiento de un control de acceso con las siguientes características:

- Escenario 1 (imágenes con fondo blanco).
- Utilización de dos componentes de PCA.
- Vecino más cercano como clasificador.

```
>> escribe_arff('A2.arff', 'A2.mat')
>> !java weka.classifiers.lazy.IBk -t A2.arff
```

Del resultado que se mostrará en pantalla, nos fijaremos en los resultados obtenidos en validación cruzada:

```
=== Stratified cross-validation ===

Correctly Classified Instances         109   90.8333 %
Incorrectly Classified Instances        11    9.1667 %
```

Los resultados indican que el sistema tendrá previsiblemente un porcentaje de aciertos del 90.8%

También nos fijaremos en la matriz de confusión obtenida en los experimentos de validación cruzada:

```
=== Confusion Matrix ===

  a  b  c  d  e  f  g  h  i  j  k  l   <-- classified as
  7  0  0  0  0  0  0  0  0  0  3  0 |  a = 01
  0 10  0  0  0  0  0  0  0  0  0  0 |  b = 02
  0  0 10  0  0  0  0  0  0  0  0  0 |  c = 03
  0  0  0  6  0  0  0  0  3  0  0  1 |  d = 04
  0  0  0  0 10  0  0  0  0  0  0  0 |  e = 05
  0  0  0  1  0  9  0  0  0  0  0  0 |  f = 06
  0  0  0  0  0  0 10  0  0  0  0  0 |  g = 07
  0  0  0  0  0  0  0 10  0  0  0  0 |  h = 08
  0  0  0  2  0  0  0  0  8  0  0  0 |  i = 09
  0  0  0  0  0  0  0  0  0 10  0  0 |  j = 10
  1  0  0  0  0  0  0  0  0  0  9  0 |  k = 11
  0  0  0  0  0  0  0  0  0  0  0 10 |  l = 12
```

La matriz de confusión indica que el sujeto 4 es el que peor se clasifica (sólo 6 aciertos de 10 intentos).

**A ENTREGAR: EJERCICIO NÚMERO 1**

**Siguiendo el procedimiento descrito anteriormente, se deberá comparar el funcionamiento de diversos clasificadores y números de componentes para los distintos escenarios del problema. El objetivo es encontrar el óptimo para cada escenario.**

**Para cada escenario, se deberán comparar los resultados obtenidos con cada clasificador y cada número de componentes de PCA.**

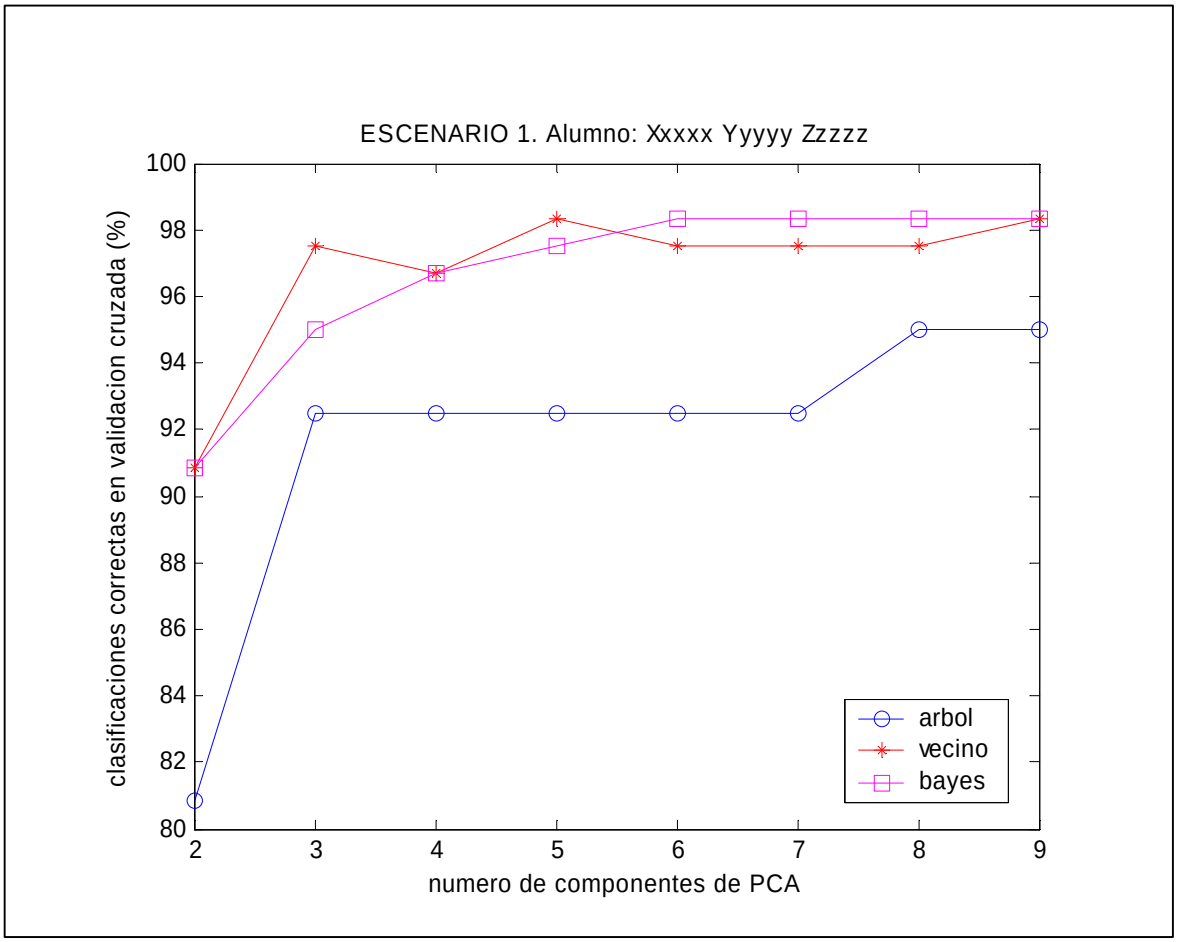
**Se considerarán todos los posibles números de componentes (de 2 a 9).**

**Y se utilizarán los siguientes algoritmos de aprendizaje:**

- **Árboles de decisión**
- **Vecino más cercano**
- **Naive Bayes**

**(todos ellos con los valores por defecto para sus parámetros, no se especificará ningún parámetro salvo el fichero de entrenamiento).**

**Los resultados se deben mostrar en tres gráficos (uno por cada escenario) que deben tener un aspecto similar al que se muestra en la figura siguiente. Lógicamente, al aumentar la dificultad del escenario, los resultados deben empeorar.**


ESCENARIO 1. Alumno: Xxxxx Yyyyy Zzzzz
clasificaciones correctas en validacion cruzada (%)
numero de componentes de PCA
arbol
vecino
bayes
100
98
96
94
92
90
88
86
84
82
80
2
3
4
5
6
7
8
9