PER (E.T.S. de Ingeniería Informática)
Curso 2018-2019

Práctica 1. Aplicación de PCA y LDA a OCR

Jorge Civera Saiz, Carlos D. Martínez Hinarejos
Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València

Índice

1. Objetivos

Los objetivos de esta práctica son los siguientes:

- Implementar las técnicas de reducción de dimensionalidad PCA y LDA estudiadas en teoría.
- Visualizar los vectores de proyección PCA y LDA.
- Realizar proyecciones lineales de los datos utilizando las matrices de proyección obtenidas mediante PCA y LDA.
- Estudiar el efecto que tienen las técnicas de reducción de dimensionalidad PCA y LDA sobre la tasa de error de un clasificador basado en vecinos más cercanos.

2. Trabajo previo a la sesión de prácticas

El presente boletín debe leerse previamente a la asistencia a la práctica. Para realizar el trabajo de esta práctica, que ocupa cuatro sesiones, se asume que el alumno conoce cómo realizar las siguientes operaciones en Octave:

- Operaciones básicas con vectores y matrices
- Calcular la media de un conjunto de vectores
- Calcular la matriz de covarianzas de un conjunto de vectores
- Calcular los eigenvectores de una matriz dada
- Creación de funciones, ficheros ".m"

Durante el desarrollo de la práctica, se aconseja tener a mano el manual de Octave: `http://www.gnu.org/software/octave/doc/interpreter/`

3. Descripción y carga de datos

Se propone realizar un clasificador por los k-vecinos más cercano de dígitos manuscritos (OCR). Para ello a modo de ejemplo se dispone de una base de datos de OCR con 7291 imágenes de entrenamiento (training) y 2006 imágenes de evaluación (test) para evaluar los diferentes parámetros del preproceso y clasificación. Los dígitos ya están escalados y centrados adecuadamente por lo que nos vamos a centrar únicamente en la reducción de la dimensionalidad.

Las imágenes de OCR son de 16×16 pixeles. Las imágenes de entrenamiento y test se encuentran en dos ficheros: `trdata.mat.gz` y `tedata.mat.gz` respectivamente. Los ficheros están en formato *ascii* comprimido para Octave. Si examinamos por ejemplo `trdata.mat.gz` veremos que contiene una cabecera como esta:

```
# name: X
# type: matrix
# rows: 256
# columns: 7291
```

Donde `#name` indica el nombre de la matriz Octave en la que se cargarán los datos, `#type` es el tipo de variable, `#rows` el número de filas y finalmente `#columns` es el número de columnas. Por lo tanto el fichero `trdata.mat.gz` contiene una matriz $X_{256\times7291}$.

En Octave cargaremos esta matriz con la siguiente orden:

```
octave:1> load trdata.mat.gz
```

Comprobaremos que la variable X se ha cargado así como su tamaño:

```
octave:2> size(X)
ans =

    256   7291
```

Cada una de las 7291 columnas de la matriz se corresponde con un vector. Dicho vector es una imagen de 16×16 pixeles. De igual forma podríamos cargar el fichero de datos de test `tedata.mat.gz`.

Dado que el objetivo es evaluar la clasificación disponemos de las etiquetas de clase de cada uno de los vectores, tanto de entrenamiento como de test. Estas etiquetas están en los ficheros `trlabels.mat.gz` y `telabels.mat.gz`, respectivamente. Cargar todos estos ficheros y comprobad en qué variables y con qué tamaño se han cargado.

4. Visualización de los dígitos

Con la notación propuesta (vectores en columnas) cada una de las imágenes de entrenamiento están en cada una de las columnas de X. Por ejemplo la imagen 100 de entrenamiento sería:

```
octave:3> x=X(:,100);
```

que quedaría almacenada en la variable vector x.

Podríamos ver dicha imagen con la orden `imshow` de Octave. Para ello debemos convertir el vector en una matriz. Dado que sabemos que originalmente cada vector proviene de una imagen de 16×16 píxeles, podemos reescribir el vector como una matriz de dicho tamaño mediante la orden `reshape` para posteriormente mostrar la matriz como una imagen:

```
octave:4> x=X(:,100);
octave:5> xr=reshape(x,16,16);
octave:6> imshow(xr',[])
```

Debería aparecer la siguiente imagen:

5. Ejercicios propuestos

Se propone implementar dos proyecciones lineales, PCA y LDA. Para ello previamente se deberían conocer la función `eig`:

`http://www.gnu.org/software/octave/doc/interpreter/Basic-Matrix-Functions.html`

La función `eig` es la función necesaria para poder obtener las proyecciones PCA y LDA. Esta función calcula los valores (y opcionalmente los vectores propios) de una matriz, o los valores propios generalizados (y opcionalmente los vectores propios generalizados) de un par de matrices (ver `help eig`). No se garantiza que los vectores propios que devuelve la función `eig` estén ordenados de mayor a menor por valor propio asociado. Por ello, será necesaria la utilización de la función `sort` (ver `help sort`).

5.1. Cálculo y proyección con PCA

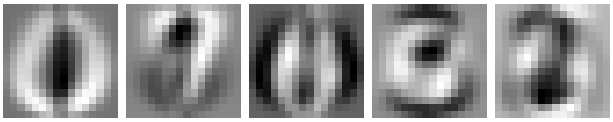
En este ejercicio el objetivo es calcular la matriz de proyección lineal PCA. Asimismo se propone representar los primeros eigenvectores para comprobar la correcta implementación.

Actividad a realizar: Se debe implementar la siguiente función:

```
function [m,W]=pca(X)
  %% Aquí el código necesario
endfunction
```

donde `X` es la matriz con los vectores en el espacio original. Como resultado obtenemos la media de los vectores `m` y la matriz de proyección `W` que contiene por filas los vectores propios ordenados de mayor a menor valor propio asociado. Esta función debe guardarse en un fichero que se llame `pca.m`.

Actividad a realizar: Para comprobar la correcta implementación de PCA se propone representar gráficamente los 5 primeros eigenvectores. Recuerda que la visualización de vectores fue descrita en la Sección 4. Los 5 primeros eigenvectores tienen la siguiente apariencia:



5.2. Calculo y proyección con LDA

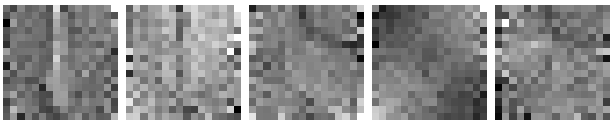
En este ejercicio el objetivo es calcular la matriz de proyección lineal LDA. Asimismo se propone representar como imagen algunos eigenvectores para comprobar la correcta implementación.

Actividad a realizar: Se debe implementar la siguiente función:

```
function [W]=lda(X,xl)
  %% Aquí el código necesario
endfunction
```

donde `X` es la matriz con los vectores en el espacio original y `xl` es el vector con las etiquetas de clase de cada uno de los vectores de entrenamiento. Como resultado obtenemos la matriz de proyección `W` que contiene por filas hasta un máximo de $C-1$ vectores propios generalizados ordenados de mayor a menor valor propio generalizado. Esta función debe guardarse en un fichero que se llame `lda.m`.

Actividad a realizar: Al igual que en PCA puedes comprobar la correcta implementación de LDA representando los 5 primeros eigenvectores:



5.3. Clasificación con proyecciones PCA y LDA

En esta sección vamos a estudiar el efecto que tienen las técnicas de reducción de dimensionalidad PCA y LDA implementadas, sobre la tasa de error de un clasificador basado en vecinos más cercanos.

Para ello se dispone en PoliformaT de dos ficheros necesarios para el uso de este clasificador: `knn.m` y `pdist2.m`. Estos dos ficheros deben descargarse al mismo directorio donde se encuentren las implementaciones de PCA y LDA.

El fichero `knn.m` contiene la siguiente función:

```
function [err]=knn(X,xl,Y,yl,KNN)
....
endfunction
```

que implementa un clasificador por vecinos más cercanos basado en distancia Euclídea. Más concretamente, esta función devuelve el error de clasificación de este clasificador usando `X` como muestras de entrenamiento (por columnas) e `Y`, como muestras de test (por columnas). Las etiquetas de clase de entrenamiento y test almacenadas en `xl` y `yl`, respectivamente, se proporcionan tal cual se cargan de fichero.

Finalmente, el parámetro `KNN` indica que para la clasificación de cada muestra de test se considerarán las `KNN` muestras de entrenamiento más cercanas en distancia Euclídea a esta muestra de test. De forma que la clase mayoritaria entre las `KNN` muestras de entrenamiento más cercanas define la clase de la muestra de test. Para los experimentos de esta práctica utilizaremos `KNN=1`.

El fichero `pdist2.m` incluye una función auxiliar que implementa el cálculo de distancias entre dos conjuntos de vectores y que se invoca desde la función `knn`.

Actividad a realizar: Para estudiar el comportamiento del clasificador basado en el vecino más cercano (`KNN`=1) en combinación con PCA se debe realizar la Figura 1.

Como se puede observar hay dos curvas en la Figura 1, la curva *Original* que representa la tasa de error del clasificador con los datos originales (256 dimensiones), y la curva *PCA* que muestra la evolución de la tasa de error en función del número de dimensiones a las cuales los datos son proyectados (`k=[10:10:100]`).

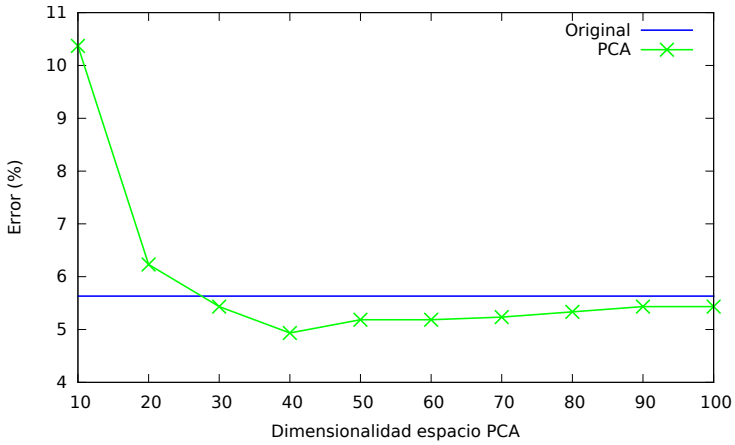


Figura 1: Resultados comparativos entre PCA y espacio original.

Actividad a realizar: Para comparar el comportamiento del clasificador de vecino más cercano dependiendo de la técnica de proyección previamente utilizada se debe realizar la Figura 2.

Al igual que en la Figura 1, se estudia la tasa de error en función del número de dimensiones a las cuales se proyecta. Sin embargo, dada la limitación de LDA a proyectar como máximo a $C-1$ dimensiones, la comparación se realiza en el rango de `k=[1:9]` dimensiones. Aunque las tasas de error obtenidas por LDA en la Figura 2 son menores que las de PCA en este rango, no se debe olvidar que PCA obtiene tasas de error menores en torno a 40 dimensiones.

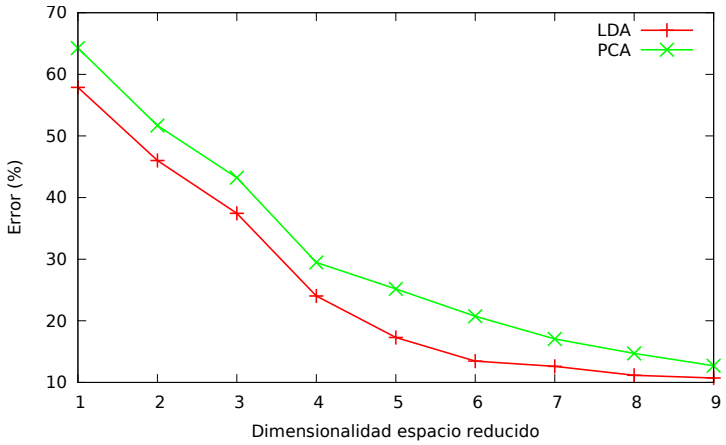


Figura 2: Resultados comparativos entre PCA y LDA.

6. Evaluación de la práctica

Esta práctica supone un 15 % del total de la nota de la asignatura (1.5 puntos). Para su evaluación se tendrán en cuenta dos actividades:

1. Una memoria del trabajo básico pedido en esta práctica, con los siguientes contenidos:

   - Código comentado de `pca.m` y `lda.m` (0.25 puntos cada uno)
   - Las dos gráficas de resultados comparativos de este boletín (0.25 puntos)
   - Comentario de los resultados obtenidos en las dos gráficas (0.25 puntos)

   Se podrá entregar como máximo hasta el día **29 de abril** a través de PoliformaT.

2. Una "competición" cuyo objetivo será lograr la mínima tasa de error del clasificador de vecino más cercano proporcionado previa aplicación de las técnicas de reducción de dimensionalidad estudiadas en clase (PCA, LDA ó PCA+LDA). Esta competición se realizará durante la última sesión de esta práctica (**16 de abril**) utilizando un corpus de datos diferente. La nota será un máximo de 0.5 puntos dependiendo del resultado obtenido.