Biometría: Ejercicios

Jorge Juan González

21 de mayo de 2022

Índice

1. Ejercicio 1: Curva ROC

1.1. Implementación

1. **Ratios:** Concatenamos y ordenamos los scores de impostores y clientes. Obtenemos la lista de umbrales como valores únicos de la lista de scores. Por cada umbral, calculamos su FPR y FNR.

2. **Curva ROC:** Para obtener la curva ROC ploteamos los ejes X e Y donde el eje X será la lista de FPR para cada umbral, y el eje Y será la lista de 1-FNR para cada umbral. A los valores de los ejes X e Y se les añade un 0 y un 1 para evitar que las gráficas queden incompletas y se mantengan dentro del rango [0,1] en ambos ejes.

3. **FP(FN=X) y umbral:** En primer lugar obtenemos las apariciones del valor X en la lista de FNRs para cada umbral. Si no hay ninguna aparición se devuelve el índice donde se encuentra el FNR más cercano a X. Si sólo hay una se devuelve el índice del elemento que cumple que FNR=X. Si hay varias apariciones, se devuelve el índice mayor si queremos una medida optimista (menos FP) o el menor si queremos una medida pesimista (más FP).
Con este índice, dado que hay tantos FPR y FNR como umbrales, podemos obtener el FPR y umbral para un cierto FNR.

4. **FN(FP=X) y umbral:** En este caso seguimos la misma lógica que en el punto anterior pero buscando apariciones del valor X en la lista de FPRs para cada umbral. De nuevo, se obtiene un índice con el que podemos obtener FNR y umbral para un cierto FPR.

5. **AUC:** El área bajo la curva se calcula como:

$$AUC = \frac{1}{C*I} * \sum_{c=1}^{C} \sum_{i=1}^{I} H(c,i)$$

$$H(c,i) = \begin{cases} 1, & \text{if } c > i \\ 0{,}5, & \text{if } c = i \\ 0, & \text{if } c < i \end{cases}$$

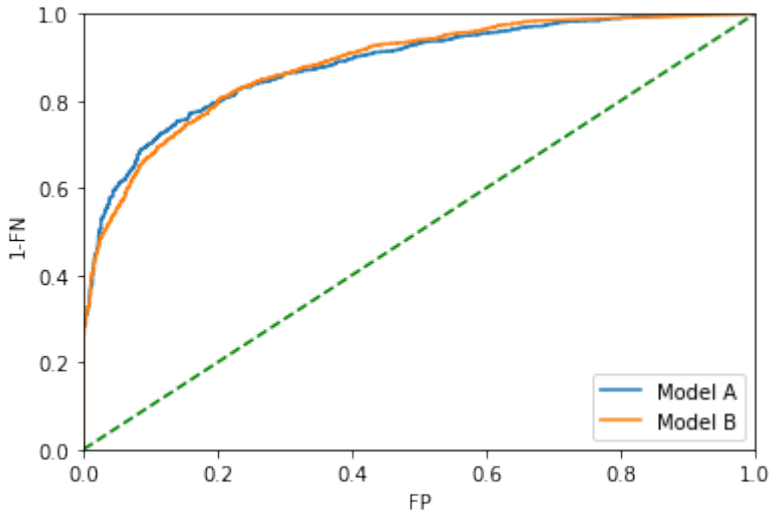
6. **D-Prime:** El factor D-Prime se calcula como:

$$d' = \frac{\mu_{clients} - \mu_{impostors}}{\sqrt{\sigma^2_{clients} + \sigma^2_{impostors}}}$$

1.2. Resultados

Las curvas ROC obtenidas para ambos modelos se muestran superpuestas a continuación. Se puede observar que ambas curvas son muy parecidas, indicando que ambos modelos tienen una calidad similar.

Figura 1: Curvas ROC obtenidas para cada modelo.



Las siguientes tablas muestran los valores devueltos por las distintas funciones implementadas para cada uno de los modelos.

Cuadro 1: Resultados obtenidos para el modelo A.

Función	**Resultado**
FP(FN=0.2)	FP=0.203205 @ thr=0.050144
FN(FP=0.2)	FN=0.202098 @ thr=0.050524
FP=FN	FP=0.201282, FN=0.201399 (Diff=0.000117) @ thr=0.050333
AUC	0.883163
D-Prime	0.759688

Cuadro 2: Resultados obtenidos para el modelo B.

Función	**Resultado**
FP(FN=0.2)	FP=0.201282 @ thr=0.04431
FN(FP=0.2)	FN=0.205594 @ thr=0.045244
FP=FN	FP=0.201282, FN=0.200699 (Diff=0.000583) @ thr=0.044359
AUC	0.883083
D-Prime	0.873177

Podemos tomar AUC como métrica de calidad a la hora de comparar varios modelos. Cuanto más se aproxime la AUC de un modelo a 1, más próximo estará este modelo a ser perfecto (ideal). Si se diera ese caso, para cualquier umbral, no habría ni falsos positivos ni falsos negativos.

Por otro lado, el factor d' nos indica la discriminalidad del modelo, es decir, cuán solapados están los scores de clientes e impostores.

Sabiendo esto, podemos concluir que el modelo A (AUC=0.883163) es ligeramente mejor que el modelo B (AUC=0.883083) aunque el modelo B (d'=0.873177) tiene mejor discriminalidad frente al modelo A (d'=0.759688).

2. Ejercicio 2: PCA

2.1. Implementación

- **Preprocesamiento:** Las imágenes son leídas con la librería OpenCV a un canal dado que son imágenes en escala de grises. Cada imagen se convierte a una fila de píxeles y se almacena como una fila en la matriz de datos de entrenamiento (X_train) y test (X_test), siendo ambos de tamaño $n \times d$. Para cada imagen se guarda su clase como el nombre de la carpeta que la contiene, la cual identifica a cada usuario del dataset, obteniendo así (Y_train) e (Y_test).

- **Eigenfaces:** La matriz de eigenvectores (B) de tamaño $d \times n$ se calcula como sigue:

  1. Se transpone la matriz de muestras (X) pasando a ser $d \times n$.
  2. Se calcula $A = X - \mu$ siendo μ de tamaño $d \times 1$.
  3. Se calcula $C' = \frac{1}{d} A^t A$
  4. Se calculan los eigenvalores y eigenvectores (B') a partir de C'.
  5. Se calcula $B = \frac{AB'}{|AB'|}$

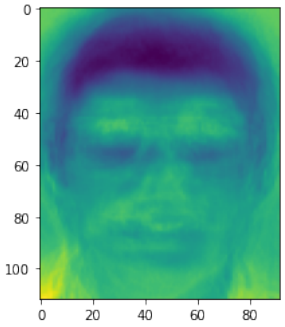
- **Proyección:** Las muestras proyectadas en k dimensiones (X_{PCA}) se calculan como sigue:

  1. Se calcula $A = X - \mu$ siendo μ de tamaño $1 \times d$.
  2. Se calcula $X_{PCA} = AB_k$ siendo B_k las primeras k columnas de B.

2.2. Resultados

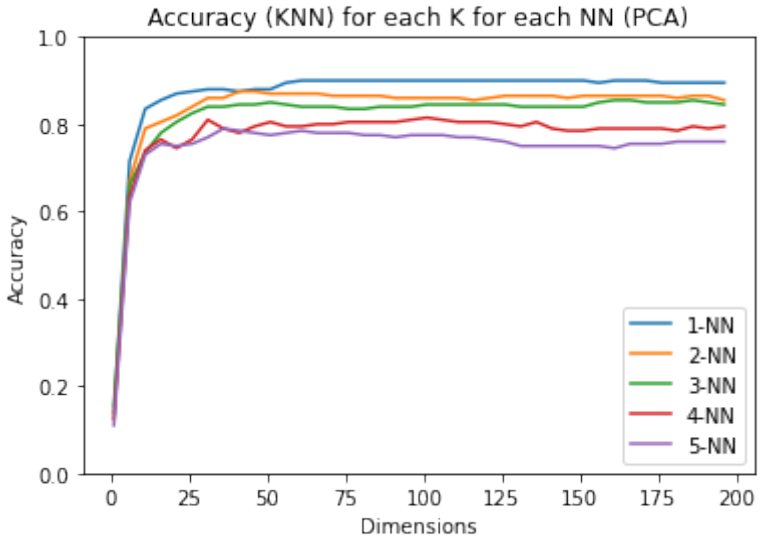
Si convertimos una columna de la matriz B a una matriz de las mismas dimensiones que las imágenes del dataset obtenemos eigenfaces como el que se muestra a continuación.

Figura 2: Eigenface.



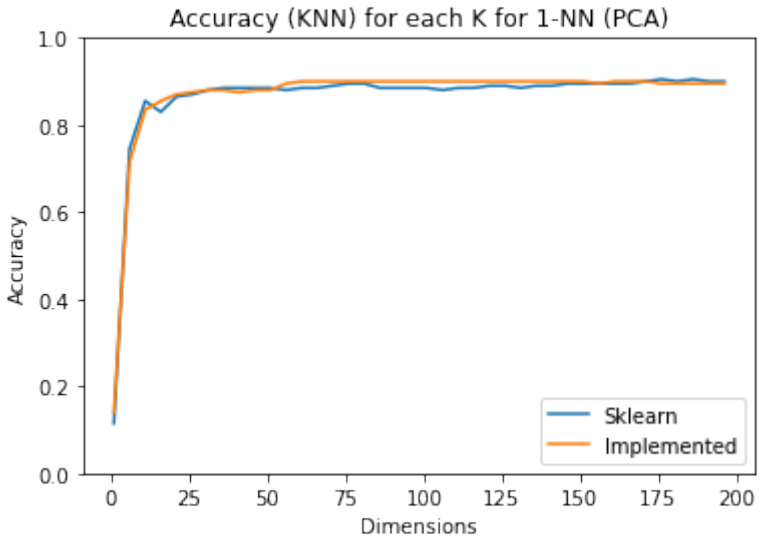
Para la evaluación hemos utilizado el clasificador KNN de la librería sklearn con el que hemos generado las curvas de error para distintas dimensiones (k). Al comparar la precisión obtenida para distinto número de vecinos más próximos (NN) obtenemos los siguientes resutlados. A mayor número de NN peor precisión, por lo que utilizaremos NN=1 para el resto de evaluaciones.

Figura 3: Comparativa del error para distintos NN.



Al comparar la precisión obtenida con nuestra implementación respecto a la obtenida con el PCA de la librería sklearn obtenemos los resultados que se muestran en la figura. Como vemos, la precisión obtenida es muy próxima a la conseguida con la librería.

Figura 4: Comparativa del error respecto al PCA de sklearn.



La mejor precisión con sklearn se consigue con 176 dimensiones llegando al 90,5 % de precisión.

La mejor precisión con nuestra implementación se consigue con 116 dimensiones llegando al 90,5 % de precisión.

3. Ejercicio 3: LDA

3.1. Implementación

- **Preprocesamiento:** Reducimos las muestras con nuestra implementación de PCA para el número de dimensiones que maximizaba la precisión en el ejercicio anterior (k_{PCA}=116). Reutilizamos Y_train e Y_test dado que no varían.
- **Obtención de B:** La matriz de eigenvectores (B_{LDA}) de tamaño $k_{PCA} \times k_{PCA}$ se calcula como sigue:
  1. Se transpone la matriz de muestras reducidas (X_{PCA}) pasando a ser $k_{PCA} \times n$.
  2. Se calcula S_b y S_w.
  3. Se calcula $C = S_w^{-1} S_b$
  4. Se calculan los eigenvalores y eigenvectores (B) a partir de C.
  5. Se ordenan los eigenvectores de mayor a menor de acuerdo con sus eigenvalores.
- **Proyección:** Las muestras proyectadas en k_{LDA} dimensiones (X_{LDA}) se calculan como $X_{LDA} = X_{PCA} B_{k_{LDA}}$ siendo $B_{k_{LDA}}$ las primeras k_{LDA} columnas de B_{LDA}.

3.2. Resultados

De nuevo, hemos evaluado la precisión usando el clasificador KNN de sklearn utilizando NN=1.

La siguiente figura muestra la curva de la precisión obtenida para distintos valores de k_{LDA} comprendidos entre 1 y k_{PCA}. A partir de alrededor de 10 dimensiones la precisión se mantiene estable aunque alcanza su máximo con 73 dimensiones manteniéndose en una precisión del 90,5 %.

Figura 5: Comparativa de la precisión obtenida para distintas dimensiones en LDA.