TRABAJO PRACTICO APRENDIZAJE PROFUNDO PARA EL PROCESAMIENTO DE LA SEÑAL BIOMÉTRICA

Alumnos:

- Rocío Rico Sanchez-Mateos
- Iker Villegas Labairu

Antes de comenzar vamos a importar todos los paquetes que vamos a utilizar a lo largo del trabajo.

In [1]:
```
#import cv2
import numpy as np
#import tensorflow.keras.models as Models
#from tensorflow.keras.layers import Flatten
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
import pandas as pd
import face_recognition_functions as frf
#import os
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.metrics import accuracy_score
```

```
WARNING:tensorflow:No training configuration found in the save file, so the model was *not* compiled. Compile it manually.
```

PREPROCESAMIENTO DE LOS DATOS Y ESTUDIO PREVIO

Antes de comenzar a entrenar y evaluar los modelos, vamos a preparar y analizar los datos que utilizaremos. En primer lugar, mediante una función que hemos definido como `obtain_train_test` sacaremos las distintas matrices de entrenamienteo y test, con sus respectivas etiquetas para cada una de las tres etnias que estudiaremos: asiáticos, cucásicos y negros. Para cada conjunto de datos obtenemos los embeddings de las imágenes en formato array siendo el 50% correspondientes a hombres y el otro 50% a mujeres.

En particular tomaremos 1000 imágenes para cada una de las étnias, ya que así se requerirá para los posteriores estudios.

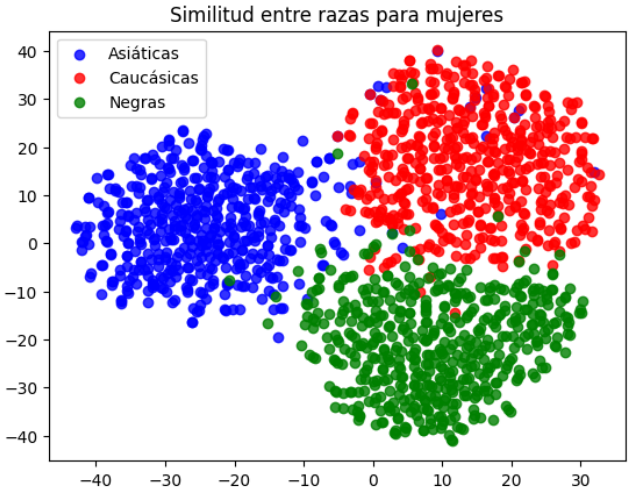
Notar que vamos a considerar por defecto que estas 1000 imágenes se corresponderán con 1000 identidades diferentes, ya que a la hora de entrenar un modelo, nos ofrecerá mejores resultados aquel en el que haya una mayor variedad de caras entrenadas.

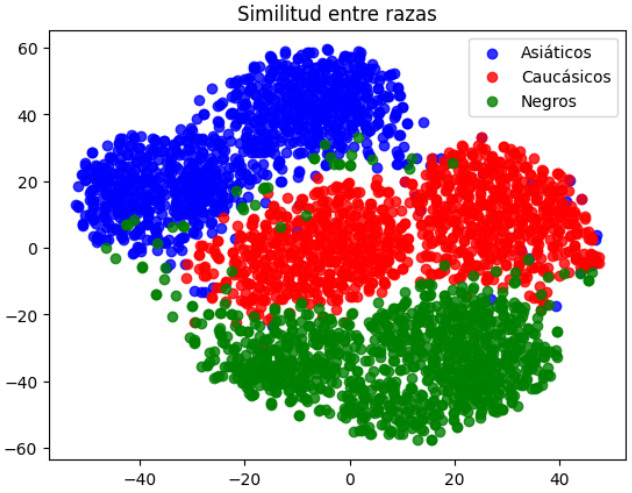
In [ ]:
```
x_train_A, y_train_A, x_test_A, y_test_A = frf.obtain_train_test(["HA4K_120","MA4K_120"])
x_train_B, y_train_B, x_test_B, y_test_B = frf.obtain_train_test(["HB4K_120","MB4K_120"])
x_train_N, y_train_N, x_test_N, y_test_N = frf.obtain_train_test(["HN4K_120","MN4K_120"])
```

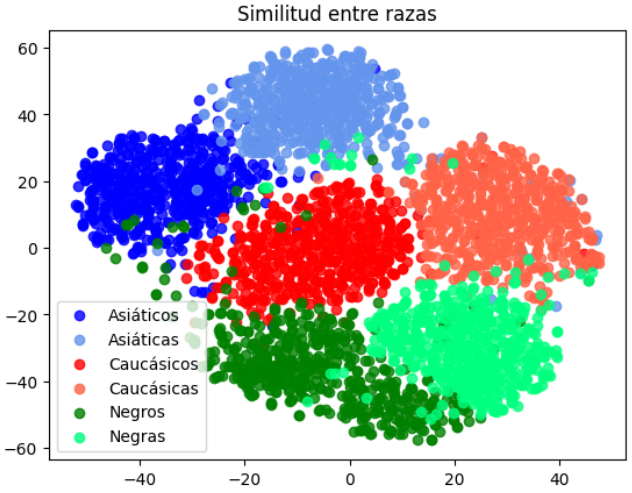
Similitud entre razas

Para poder entender mejor los análisis posteriores vamos a ver en la siguiente gráfica la similitud que hay entre razas. Realizaremos una reducción de dimensionalidad mediante el algoritmo TSNE con el objetivo de representar los embeddings de los datos de entrenamiento en dos dimensiones.

In [20]:
```
frf.representacion_TSNE(x_train_A,x_train_B,x_train_N)
```







-De la primera figura podemos afirmar que en hombres la diferencias entre etnias está muy marcada sin una aparente acercamiento de unas sobre otras. Hay tres clases bien diferenciadas pese a que los colores cerca del centro sean más heterogénos o haya outliers.

-De la segunda figura también podemos decir que hay tres clases diferenciadas pero además, hay que añadir que se aprecia una mayor similitud entre mujeres caucasicas y negras respecto de las asiaticas. Dentro de este grupo, las caucásicas se presentan más cercanas a las asiáticas que las mujeres negras.

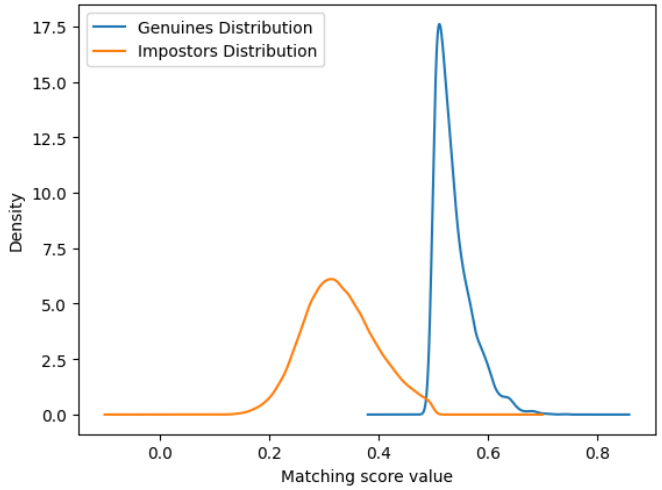
-De la tercera figura podemos observar las diferentes ideas presentadas en las anteriores gráficas: se dividen los datos en tres grupos bien diferenciados representando las tres etnias y en cada subgrupo puede apreciarse una división por las diferencias de género, la cuál podemos ver mejor mediante la figura 4. Se puede apreciar que puntos se refieren a mujeres si buscamos cuales son las mitades más próximas para caucásicos y negros como se explicaba en el punto anterior. Fuera de estas obervaciones antes mencionadas hay que destacar como el grupo asociado a la etnia asiática aparece más separada del resto, en particular las asiáticas. Esto puede deberse a que dispongan de rasgos o características que las diferencien con mayor facilidad del resto de etnias.

Curvas FAR y FRR de los datos a evaluar

Con el objetivo de ver las características que encierran los datos que vamos a evaluar con posteridad, vamos a ver el comportamiento de sus curvas FAR y FRR.

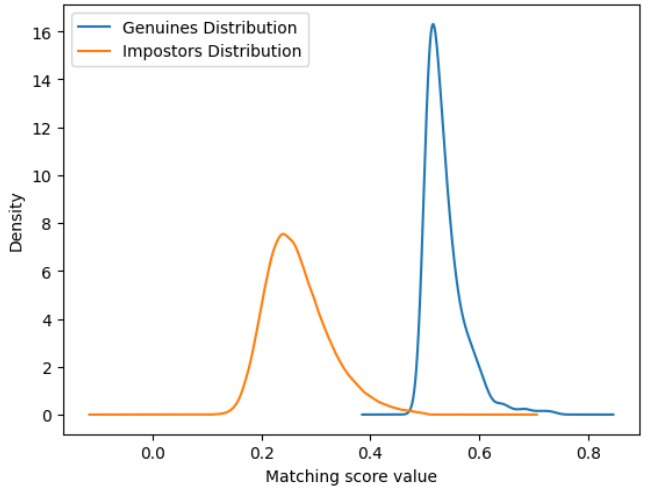
- Comencemos con el caso de los asiáticos

In [3]:
```
score_genuino_A, score_impostor_A = frf.get_scores(x_test_A)
frf.curva_FAR_FRR(score_genuino_A,score_impostor_A)
```



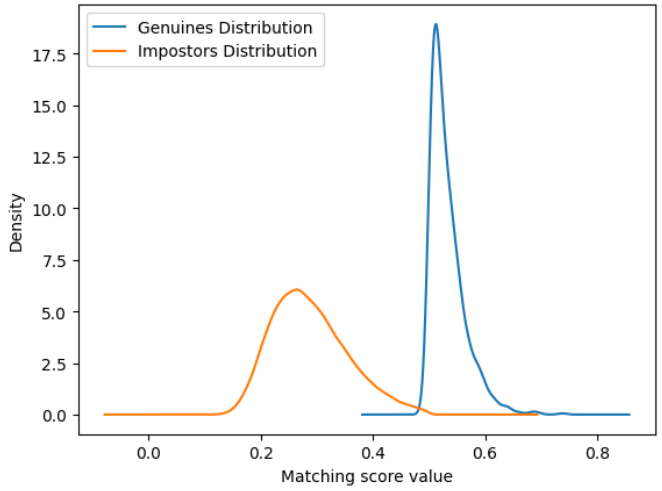
- Continuemos con la etnia de los caucásicos

In [4]:
```
score_genuino_B, score_impostor_B = frf.get_scores(x_test_B)
frf.curva_FAR_FRR(score_genuino_B,score_impostor_B)
```



- Por último realicemos las curvas para el caso de la etnia negra

In [5]:
```
score_genuino_N, score_impostor_N = frf.get_scores(x_test_N)
frf.curva_FAR_FRR(score_genuino_N,score_impostor_N)
```



Para las tres etnias, podemos ver como la cantidad de scores genuinos tiene muy poca variabilidad, tal y como se puede apreciar por la delgadez de su curva gaussiana. Sin embargo, los scores impostores presentan una variabilidad mayor. Esto nos permite concluir que todas las imágenes presentan un comportamiento de similitud muy parecido, lo cuál es evidente ya que todas las etnias presentan rasgos similares característicos: como por ejemplo el pelo negro propio de los asiáticos, el pelo corto en mujeres negras, o más claramente, el color de la piel propio de cada raza. Mientras que en el rango de impostores se encontraran aquellos datos de test que sean propios de características no muy propias de la raza, como asiáticos con otra coloración de pelo o mujeres negras que presenten pelo largo, entre otros casos.

TASK 2.1: Entrenar 3 clasificadores de género diferentes usando imágenes del mismo grupo étnico

Para llevar el entrenamiento a cabo, vamos a crear 3 clasificadores los cuales entrenaremos usando únicamente imágenes de uno de los 3 grupos étnicos para cada uno: *estimator_modelA* el correspondiente a los asiáticos, *estimator_modelB* para los caucásicos y *estimator_modelN* el cuál será entrenado usando únicamente imágenes de personas negras.

In [7]:
```
# Se crea un modelo para el clasificador
#input_size = 2048
def baseline_model():
    model = Sequential()
    model.add(Dense(60, input_dim=2048, activation='relu'))
    model.add(Dense(2, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# Se crea el clasificador
estimator_modelA = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
estimator_modelB = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
estimator_modelN = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_2912\2109091981.py:11: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for help migrating.
  estimator_modelA = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_2912\2109091981.py:12: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for help migrating.
  estimator_modelB = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_2912\2109091981.py:13: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for help migrating.
  estimator_modelN = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
```

Una vez creados los clasificadores, vamos a entrenarlos usando los conjuntos de datos de entrenamiento que hemos generado con anterioridad. Para cada uno de ellos se utilizarán 1000 imágenes: 500 correspondientes a hombres y la otra mitad a mujeres.

In [8]:
```
# Entrenamiento del clasificador
estimator_modelA.fit(x_train_A, y_train_A) #grupo asiático
estimator_modelB.fit(x_train_B, y_train_B) #grupo caucásico
estimator_modelN.fit(x_train_N, y_train_N) #grupo negro
```

Out[8]:
```
<keras.callbacks.History at 0x11a1d35ae60>
```

Y con esto ya tenemos creados y entrenados nuestros tres clasificadores.

TASK 2.2: Evaluar los tres clasificadores de género usando imágenes de cada uno de los tres grupos étnicos.

Para llevar a cabo esta segunda tarea, evaluaremos cada uno de los tres clasificadores usando 3 conjuntos de 500 imágenes test de cada uno de los grupos étnicos, donde mitad son mujeres y la otra mitad hombres.

Modelo A: entrenado a base de imágenes de etnia asiática.

Vamos a sacar la precisión evaluando el modelo con imágenes de las 3 etnias consideradas.

In [9]:
```
# Predicción
y_pred_A = estimator_modelA.predict(x_test_A) # asiáticos
y_pred_A = np_utils.to_categorical(y_pred_A)
y_pred_B = estimator_modelA.predict(x_test_B) # blancos
y_pred_B = np_utils.to_categorical(y_pred_B)
y_pred_N = estimator_modelA.predict(x_test_N) # negros
y_pred_N = np_utils.to_categorical(y_pred_N)

# Resultados
score_AA = accuracy_score(y_test_A, y_pred_A)
score_AB = accuracy_score(y_test_B, y_pred_B)
score_AN = accuracy_score(y_test_N, y_pred_N)
```

```
16/16 [==============================] - 0s 4ms/step
16/16 [==============================] - 0s 2ms/step
16/16 [==============================] - 0s 2ms/step
```

In [10]:
```
print("La predicción para asiáticos entrenando con asiáticos tiene un escore de: ",score_AA)
print("La predicción para caucásicos entrenando con asiáticos tiene un escore de: ",score_AB)
print("La predicción para negros entrenando con asiáticos tiene un escore de: ",score_AN)
```

```
La predicción para asiáticos entrenando con asiáticos tiene un escore de:  0.99
La predicción para caucásicos entrenando con asiáticos tiene un escore de:  0.85
La predicción para negros entrenando con asiáticos tiene un escore de:  0.942
```

Modelo B: entrenado a base de imágenes de etnia caucásica.

Ahora repitamos el procedimiento anterior utilizando el modelo entrenado con imágenes procedentes del grupo caucásico.

In [11]:
```
# Predicción
y_pred_A = estimator_modelB.predict(x_test_A) # asiáticos
y_pred_A = np_utils.to_categorical(y_pred_A)
y_pred_B = estimator_modelB.predict(x_test_B) # blancos
y_pred_B = np_utils.to_categorical(y_pred_B)
y_pred_N = estimator_modelB.predict(x_test_N) # negros
y_pred_N = np_utils.to_categorical(y_pred_N)

# Resultados
score_BA = accuracy_score(y_test_A, y_pred_A)
score_BB = accuracy_score(y_test_B, y_pred_B)
score_BN = accuracy_score(y_test_N, y_pred_N)
```

```
16/16 [==============================] - 0s 3ms/step
16/16 [==============================] - 0s 2ms/step
16/16 [==============================] - 0s 2ms/step
```

In [12]:
```
print("La predicción para asiáticos entrenando con caucásicos tiene un escore de: ",score_BA)
print("La predicción para caucásicos entrenando con caucásicos tiene un escore de: ",score_BB)
print("La predicción para negros entrenando con caucásicos tiene un escore de: ",score_BN)
```

```
La predicción para asiáticos entrenando con caucásicos tiene un escore de:  0.848
La predicción para caucásicos entrenando con caucásicos tiene un escore de:  0.986
La predicción para negros entrenando con caucásicos tiene un escore de:  0.914
```

Modelo N: entrenado a base de imágenes de etnia negra.

Por último hagamos lo propio para el último modelo correspondiente al grupo de etnia negra.

In [13]:
```
# Predicción
y_pred_A = estimator_modelN.predict(x_test_A) # asiáticos
y_pred_A = np_utils.to_categorical(y_pred_A)
y_pred_B = estimator_modelN.predict(x_test_B) # caucásicos
y_pred_B = np_utils.to_categorical(y_pred_B)
y_pred_N = estimator_modelN.predict(x_test_N) # negros
y_pred_N = np_utils.to_categorical(y_pred_N)

# Resultados
score_NA = accuracy_score(y_test_A, y_pred_A)
score_NB = accuracy_score(y_test_B, y_pred_B)
score_NN = accuracy_score(y_test_N, y_pred_N)
```

```
16/16 [==============================] - 0s 2ms/step
16/16 [==============================] - 0s 2ms/step
16/16 [==============================] - 0s 5ms/step
```

In [14]:
```
print("La predicción para asiáticos entrenando con negros tiene un escore de: ",score_NA)
print("La predicción para caucásicos entrenando con negros tiene un escore de: ",score_NB)
print("La predicción para negros entrenando con negros tiene un escore de: ",score_NN)
```

```
La predicción para asiáticos entrenando con negros tiene un escore de:  0.868
La predicción para caucásicos entrenando con negros tiene un escore de:  0.888
La predicción para negros entrenando con negros tiene un escore de:  0.96
```

La matriz de precisión de nuestros clasificadores para distinguir entre géneros es la siguiente:

$$Accuracy\ score = \begin{pmatrix} 0.990 & 0.850 & 0.942 \\ 0.848 & 0.986 & 0.914 \\ 0.868 & 0.888 & 0.960 \end{pmatrix}$$

donde cada fila de resultados ha sido entrenada con datasets siguiendo el mismo orden que el de evaluación de las columnas: asiáticos, caucásicos y negros.

Como esperábamos la diagonal es la que mayores valores posee ya que esos datos predicen sobre la misma etnia con la que han sido entrenados. A parte de eso encontramos relaciones interesantes como que aparentemente los más fáciles de clasificar según género sean las personas de raza negra independientemente de los datos de entrenamiento. Por el contrario, la etnia más difícil de distinguir es la asiática. Esto en primera instancia podría sugerir que el dimorfismo de humanos se ve más acentuado en etnias negras mientras que los asiáticos tienden hacia la unificación de los caracteres sexuales pero no hay que dejar pasar las diferencias culturales en atributos variables como el corte de pelo, maquillaje, gafas, depilado de cejas o barba, incluso la tendencia a sonreir en fotos o el ideal de peso de cada país, etc. Para corroborar estas influencias tendríamos que hacer un estudio mucho más exahustivo pudiendo aún así no dar con la verdadera causa por la cual nuestro clasificador toma las decisiones que hace. Por último comentar que el entrenamiento que mayor precisión consigue es el entrenamiento con asiáticos. Este hecho coincide con lo descrito anteriormente ya que si nuestro programa es capaz de captar las sutilezas que distinguen a unos de otros en etnias más uniformes le resultará fácil extrapolar ese conocimiento al resto, es por esto mismo, que la entrenada con personas negras devuelve los peores resultados al haberse especializado en características divergentes más obvias y tener que moverse a datos con mayor complejidad de análisis.

TASK 2.3: Entrenar un clasificador de género usando imágenes de los tres grupos étnicos.

Al igual que en el "TASK 2.1" comenzamos definiendo un nuevo clasificador para este modelo.

In [15]:
```
estimator_model_total = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_2912\2226731418.py:1: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adr
iangb/scikeras) instead. See https://www.adriangb.com/scikeras/stable/migration.html for help migrating.
  estimator_model_total = KerasClassifier(build_fn=baseline_model, epochs=200, batch_size=5, verbose=0)
```

A continuación vamos a construir la matriz de datos de entrenamiento y el vector de etiquetas, concatenando los datos de train calculados previamente con las imágenes de cada grupo étnico. Con esto tendremos, a modo de entrenamiento para nuestro modelo, 1000 imágenes de asiáticos, 1000 de caucásicos y 1000 de negros, siendo la mitad hombres y la otra mitad mujeres.

In [16]:
```
x_train_total = np.concatenate((x_train_A, x_train_B, x_train_N), axis=0)
y_train_total = np.concatenate((y_train_A, y_train_B, y_train_N), axis=0)
```

Y finalmente entrenamos este modelo a partir de los datos que acabamos de construir.

In [17]:
```
# Entrenamiento del clasificador
#y_test_total = np_utils.to_categorical(np.asarray(y_total))
estimator_model_total.fit(x_train_total, y_train_total)
```

Out[17]:
```
<keras.callbacks.History at 0x11a1f513eb0>
```

TASK 2.4: Evaluar el clasificador de género previo usando imágenes de cada uno de los tres grupos étnicos.

Una vez entrenado el clasificador, vamos a realizar tres evaluaciones usando imágenes de cada grupo étnico por evaluación. Para ello, vamos a utilizar los datos de test obtenidos anteriormente. Una vez obtenida la predicción que nos da el clasificador para cada conjunto test, calcularemos la precisión comparandola con la etiqueta real.

In [18]:
```
# Predicción
y_pred_A = estimator_model_total.predict(x_test_A) # asiáticos
y_pred_A = np_utils.to_categorical(y_pred_A)
y_pred_B = estimator_model_total.predict(x_test_B) # blancos
y_pred_B = np_utils.to_categorical(y_pred_B)
y_pred_N = estimator_model_total.predict(x_test_N) # negros
y_pred_N = np_utils.to_categorical(y_pred_N)
#y_pred_total = estimator_model_total.predict(x_test_total) # total
#y_pred_total = np_utils.to_categorical(y_pred_total)

# Resultados
score_AA = accuracy_score(y_test_A, y_pred_A)
score_AB = accuracy_score(y_test_B, y_pred_B)
score_AN = accuracy_score(y_test_N, y_pred_N)
#score_total = accuracy_score(y_test_total, y_pred_total)
```

```
16/16 [==============================] - 0s 2ms/step
16/16 [==============================] - 0s 2ms/step
16/16 [==============================] - 0s 2ms/step
```

Así, las precisiones obtenidas serán las siguientes:

In [19]:
```
print("La predicción para asiáticos entrenando con las 3 etnias tiene un escore de: ",score_AA)
print("La predicción para caucásicos entrenando con las 3 etnias tiene un escore de: ",score_AB)
print("La predicción para negros entrenando con las 3 etnias tiene un escore de: ",score_AN)
#print("La predicción para todas las etnias entrenando con las 3 etnias tiene un escore de: ",score_total)
```

```
La predicción para asiáticos entrenando con las 3 etnias tiene un escore de:  0.994
La predicción para caucásicos entrenando con las 3 etnias tiene un escore de:  0.986
La predicción para negros entrenando con las 3 etnias tiene un escore de:  0.972
```

Observando los resultados, vemos que la precisión de las predicciones son mejores que las obtenidas por clasificadores entrenados para una única etnia. En particular, nos dan resultados muy buenos, bastante cercanos a 1. Comparándolos con la matriz anterior, incluso obtenemos valores iguales o mejores que en los casos de evaluaciones de datos con imágenes de la propia etnia usada para entrenar dicho modelo. Esto nos permite deducir con gran claridad, que un entrenamiento realizado con imágenes que presenten gran diversidad y variedad (en este caso una variedad de etnia), nos permiten obtener clasificadores más generales y más precisos a la hora de realizar predicciones.

Por otro lado, los *scores* obtenidos nos muestran que se obtienen predicciones más precisas para el caso de asiáticos que para los otros casos, dentro de estos buenos valores. Y el caso contrario ocurre con la etnia negra. Esto nos plantea los mismos razonamientos que hemos desarrollado anteriormente en los modelos entrenado con una única etnia.