Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Reconocimiento de parentescos de consanguinidad de primer grado a través del análisis automático de fotos

Tesis presentada para optar al título de
Licenciado en Ciencias de la Computación

Juan Manuel Ortiz de Zarate

Director: Dr. Diego Fernández Slezak
Codirector: Lic. Sergio Romano
Buenos Aires, 2016

RECONOCIMIENTO DE PARENTESCOS DE CONSANGUINIDAD DE PRIMER GRADO A TRAVÉS DEL ANÁLISIS AUTOMÁTICO DE FOTOS

La identificación automática mediante fotos de parentescos de consanguinidad de primer grado tiene como objetivo poder determinar computacionalmente cuando dos personas son parientes directos y cuando no. La intención de hacerlo así es aprovechar el poder de cómputo para analizar millones de fotos entre sí en un corto plazo y poder determinar, con una alta probabilidad, esta relación. Esta masividad de procesamiento que nos proporcion adicha técnica puede ser utilizada para el hallazgo de personas desparecidas o hijos de desaparecidos, teniendo en cuenta lo sucedido en nuestro país entre los años 1976 y 1983 bajo la última dictadura cívico-militar.
En este trabajo logramos porcentajes de aciertos muy buenos, a la hora de determinar estos parentescos, en distintas bases de imágenes con diferentes normalizaciones. A través Boosted Cascade pudimos extraer de forma automática los rasgos faciales, para después mediante algoritmos de machine learning lograr una tasa de aciertos por encima del 90 % en el mejor de los casos. En muchos otros por encima de la capacidad de detección humana de estas relaciones (67.19 %).
Además comprobamos que se generan mejores modelos utilizando imágenes de la madre en vez del padre e incluso utilizando ambos a la vez. También corroboramos que aplicarle PCA a la matriz de relaciones, previo a ejecutar el algoritmo de machine learning, mejora la homogeneización del funcionamiento de los clasificadores a través de las distintas bases de imágenes.

Palabras claves: Machine Learning, Kinship Verification, Siblings, PCA, Boosted Cascade, K-Nearest-Neighbors.

AGRADECIMIENTOS

A mis viejos y mi hermana por haberme bancado siempre
A Kevin Kujawski por haber hecho cada materia y cada TP de la carrera juntos
A todos mis compañeros de cursada y en especial a Leo Teren, Alexis Soifer y Martin Carreiro por habernos ayudado y bancado en cada cursada
A todos mis compañeros de La Campora Exactas por haberme hecho parte de este sueño de transformar nuestra facultad, la ciencia y la realidad
A Diego y Sergio por haberme traído un tema tan lindo a desarrollar en la tesis y por orientarme y ayudarme en ella
A mi abuelo por haberme inculcado esta pasión por la ciencia y la formación
A Nestor y Cristina por haberle dado lugar e importancia nuevamente a la educación, la ciencia y la patria.

A mi abuelo
A Nestor y Cristina

Índice general

1. INTRODUCCIÓN

La identificación de parentescos entre personas es una disciplina bastamente investigada y aplicada por las ciencias biológicas que nos permite saber con un altísimo grado de exactitud si dos personas son parientes directos (e indirectos también)Lévi-Strauss [1969], Carsten [2000]. Esto tiene una gran utilidad para la identificación de personas desaparecidas. Mas precisamente, en nuestro país a través del Equipo Argentino de Antropología Forense, esta disciplina y las Abuelas de Plaza de Mayo, se ha logrado encontrar a 121 hijos de desaparecidos de la última dictadura cívico-militar. Para ello, se desarrolló el *índice de abuelidad*, en colaboración con la investigadora Marie Claire King, que logró identificar con un 99,9 % de exactitud el parentesco salteando una generación utilizando métodos genéticosDi Lonardo et al. [1984]. Sabemos que aún quedan muchos nietos más por recuperar y con la intención de ayudar en esto es que nos propusimos colaborar desde la computación para detectar potenciales nietos a recuperar.

Nos proponemos atacar el problema desde el análisis de imágenes. ¿Acaso será posible detectar parentesco (con cierta probabilidad) entre dos personas analizando sus fotos? Si esto fuera posible, ayudaría en la búsqueda de posibles candidatos en bases exhaustivas de imágenes. Si bien es difícil poder decir con tanta exactitud esto desde esta área, la idea es que mediante las fotos digitales de las caras de las personas podamos determinar candidatos de parientes directos (madre/padre-hijo/a). De esta manera si esta técnica fuera efectiva, podría set utilizada por la secretaría de derechos humanos (o algún otro organismo estatal) en una gran base de imágenes de los argentinos (por ejemplo, la base de datos de DNI) y cotejar contra las bases de datos de desaparecidos. Este trabajo deja abierta la posibilidad de seguir perfeccionando la técnica y así seguir colaborando, desde la computación, con estas problemáticas. Además a partir de esto se podría seguir el trabajo para extenderlo a abuelos/as-nietos/as y poder así usar las fotos de las abuelas, salteando una generación.

1.1. Trabajos previos

Como punto de partida tomamos la base de datos de fotos de padres/madres-hijos/hijas Fang et al. [2010], donde analizan distintos atributos de las imágenes de caras para establecer si hay parentesco . En este trabajo se logra una importante efectividad a la hora de establecer este reconocimiento, con alrededor de 70 % de aciertos, que según su investigación está por encima de la capacidad humana que es 67,19 %. El primer objetivo de esta tesis, es tomar esta base de datos y replicar los resultados obtenidos. Luego, evaluaremos con distintos métodos de *Machine Learning* si es posible mejorar este rendimiento.

En segunda instancia, utilizaremos la base de datos publicada por la universidad de Northeastern, en la que se cuenta con dos imágenes de los padres (de joven y de adulto), lo cual nos provee un mayor conjunto de datos para trabajar que nos permita mejorar los modelos de predicción ya que contamos con dos imágenes de la misma persona S. Xia and Fu [2012, 2011], M. Shao and Fu [2011].

En una tercera instancia, también utilizaremos la base de datos de hermanos usada en Vieira et al. [2014]. Decidimos agregarla por la gran normalización que poseen sus imágenes en cuanto a tamaño, definición y luminosidad y para constatar con otro tipo de relación la generación de estos modelos.

Por otro lado, además de replicar los resultados obtenidos, proponemos modificar los algoritmos de extracción de atributos utilizados para clasificar parentesco. En este sentido, aplicamos uno de los métodos más utilizados, incluido en el paquete de procesamiento de imágenes de Matlab propuesto por Viola y JonesViola and Jones [2001]. Combinando los atributos propuestos por los artículos mencionados, sumado a la incorporación de estos últimos, esperamos mejorar el rendimiento de clasificación empujando estas técnicas hacia la detección de parentezco salteando generaciones.

A continuación describiremos las principales características, que nos importan para nuestro trabajo, de las bases de imágenes mencionadas.

1.1.1. Kinship Dataset

Esta base de imágenes corresponde al trabajo del cual partimos (Fang et al. [2010]). Contiene 140 pares de fotos de la cara (sin que se vea otra parte del cuerpo) de sujetos cuya relación es de parentesco de consanguinidad de primer grado y entre los sujetos de distintos pares no existe relación consanguínea alguna. Dichos pares son de distintos géneros (padre-hijo, padre-hija, madre-hijo y madre-hijo) y morfologías faciales (orientales, caucásicos y negroides). Es importante destacar que las imágenes se encuentran normalizadas en su tamaño (100px x 100px) y, en menor grado, en la posición del rostro, la idea es que estén lo mas frontales posibles pero en algunos casos esto no es tan así. La luminosidad no está normalizada, por ejemplo, en algunas fotos podemos observar que la luz esta de frente y es blanca mientras que en otros casos esta a la izquierda de la persona y es amarilla. Por último es importante notar que algunas fotos están en escala de grises en vez de RGB y que las caras no tienen la misma expresión, algunos están riéndose mientras otros estan serios.

Fig. 1.1: Imágenes de madre e hijo.

1.1.2. Kinface Dataset

Esta otra base corresponde a lo publicado en S. Xia and Fu [2012, 2011], M. Shao and Fu [2011]. Contiene 200 pares de sujetos cuya relación es de parentesco de consanguinidad de primer grado y entre los sujetos de distintos pares no existe relación consanguínea alguna. Como se dijo anteriormente hay dos fotos de cada padre, una de cuando era joven y otra actual. Las fotos están menos normalizadas que las de Kinship, además de no estar normalizadas en luminosidad, posición, gesto y color (hasta tiene algunas en sepia) no tienen todas la misma dimensión y se ven más partes del cuerpo además de la cara. Sí tienen mejor definición en comparación con Kinship, donde estas son de 100pxX100px y aquí todas superan los 200pxX200px. También posee distintos géneros y fenotipos de los sujetos, incluso hay algunos casos que se usan en ambos datasets (como Angelina Jolie y su padre).

Fig. 1.2: Imágenes de padre e hija.

1.1.3. Siblings Dataset

Esta última base corresponde al trabajo publicado en Vieira et al. [2014]. Contiene 90 pares de imágenes de sujetos que, a diferencia de los otros dos datasets, son hermanos entre sí y entre los sujetos de distintos pares no existe relación consanguínea alguna. Las fotos se encuentran sumamente normalizadas, la gran mayoría tienen un mismo fondo monocromático (verde), una misma luminosidad, dimensión, posición de la cara y expresión. Poseen además una muy buena definición (4256px x 2832px). Tienen distintos géneros pero son todos de la misma morfología facial (caucásico) y ninguno de los pares se repite con alguno de los otros datasets.

Fig. 1.3: Imágenes de hermanos.

1.1.4. Hijos de otro padre

Hasta hace poco tiempo existía un mito según el cual el 10 % de la población no era hija biológica de su padre legal, es decir el verdadero padre no era con quien se había criado y quién creía sino otro. Sin embargo los investigadores Mark A. Jobling y Chris Tyler-Smith rompieron con esta creencia en su trabajo *Fathers and sons: the Y chromosome and human evolution* Jobling and Tyler-Smith [1995], donde tomaron el cromosoma Y de 1600 hombres no relacionados con 40 apellido diferentes para ver si había una conexión entre el cromosoma y el apellido, ambos heredados del padre. Así es que descubrieron (entre otras cosas) que tan sólo 1 de cada 25 esta en esa situación, osea un 4 % y no un 10.
A partir de esta investigación es que decidimos ver si tiene alguna incidencia en nuestro trabajo. Ya que hay una probabilidad (pequeña) de que estemos tomando por positiva una relación Padre-hijo/a que en realidad no lo es. Para eso en nuestras pruebas vamos a generar los modelos con padres y madres mezclados pero también por separado.

2. PROCESO EXPERIMENTAL

En esta sección explicaremos cómo fué todo el proceso experimental. Los features seleccionados, la extracción de los mismos, el pre-procesamiento de ellos, la conformación de la matriz de pares, la creación de los modelos y la validación de los mismos.
Antes de desarrollar cada paso en detalle es importante entender el proceso en su conjunto, es decir, cómo llegamos desde nuestra base de datos a la creación del modelo y su validación. En el siguiente gráfico podemos ver, en un alto nivel, cómo es este procedimiento:

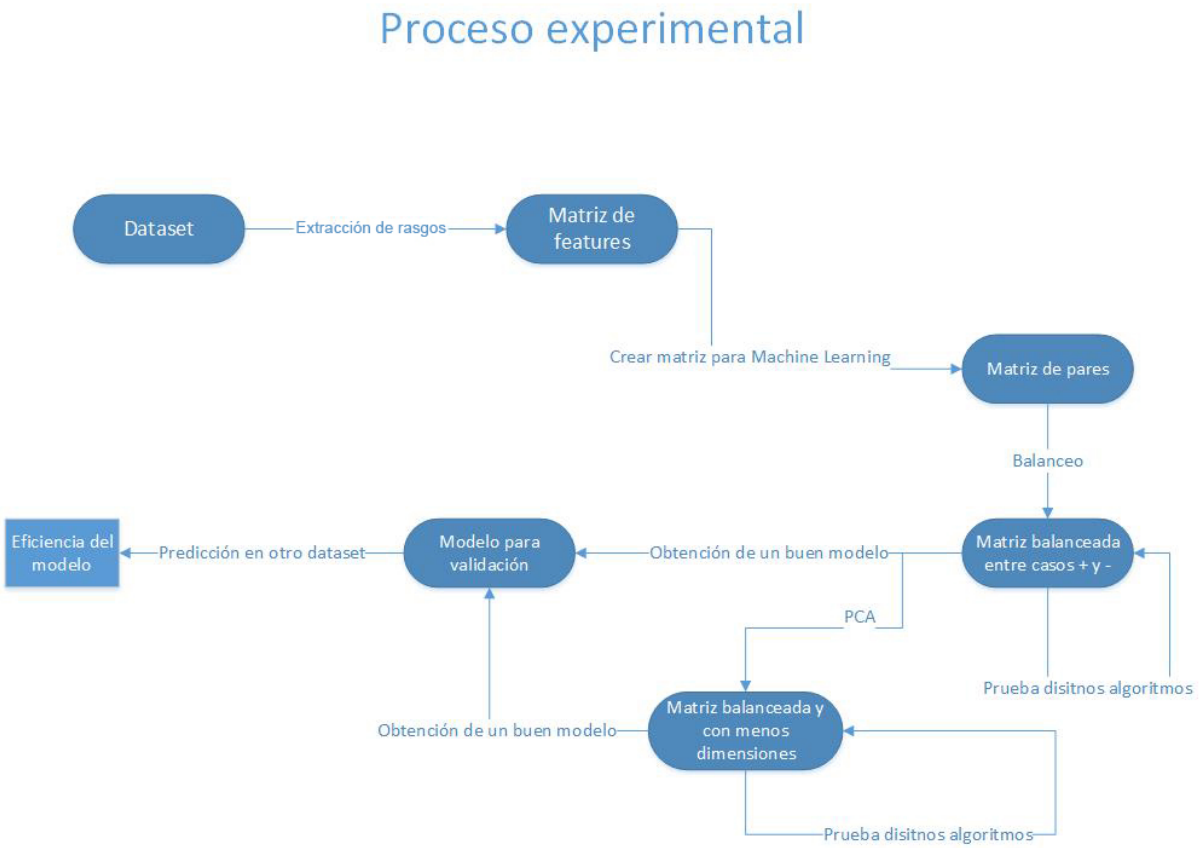

Fig. 2.1: Pasos del proceso experimental.

2.1. Selección de features

Una parte fundamental de este trabajo es poder establecer y constatar qué rasgos faciales son los que juegan un rol preponderante en la determinación de parentescos entre personas. Estudios anteriores han recolectado distintos atributos para hacer este tipo de inferencia. Por ejemplo, en Fang et al. [2010] han encontrado algunos atributos que permiten entrenar un modelo de predicción que permite clasificar pares de fotos de parientes y no parientes con una precisión de alrededor de 70 %, es decir ligeramente por encima de la capacidad humana. En la tabla 2.1 se resumen los atributos utilizados en ese estudio y sus resultados.

Rasgo	efectividad
Color ojo izquierdo separado en RGB	60.50 %
Color ojo derecho separado en RGB	61.43 %
Color de piel separado en RGB	59.70 %
Color de boca separado en RGB	56.37 %
Color de piel en escala de grises	59.83 %
Distancia entre ojos	55.47 %
Distancia entre los ojos y la boca	55.80 %

Tab. 2.1: Porcentaje de efectividad por rasgo en el trabajo Fang et al. [2010]

En este trabajo, en primer lugar nos proponemos replicar los resultados obtenidos a partir de los atributos meniconados anteriormente, y aplicarlos en los distintos datasets disponibles. Por otro lado, se propone la incorporación de nuevos atributos que provean información de características faciales, por ejemplo: el tamaño de la nariz, los ojos y la boca respecto de la cara, el tamaño de la nariz respecto de los ojos y tamaño de la boca respecto de la nariz.

2.2. Extracción de rasgos

Es importante poder extraer de forma automática estos rasgos, para de esta manera poder aprovechar el poder de cómputo y analizar muchas más imágenes en un corto plazo. Para esto primero debemos ver cómo detectar las zonas donde se encuentran la nariz, ojos, boca y cara. En su trabajo Fang Fang et al. [2010] utiliza *pictorial structure model*Felzenszwalb and Huttenlocher [2005]. Este es un procedimiento mediante el cual se genera un modelo genérico de identificación de objetos, para el cual es necesario entrenar a mano el clasificador, marcándole en un número acotado de casos (entre 5 y 15) cuales son los rasgos que identifican a ese objeto. Cada rasgo es modelado y calculado utilizando *normalizada cross-correlation* que funciona definiendo primero una imagen de plantilla que nos gustaría encontrar como una subimagen dentro de la imagen objetivo. Las plantillas consisten en un template del ojo izquierdo, ojo derecho, nariz y boca. Luego, usándolas, podemos calcular una puntuación de coincidencia en cada posición en la imagen dada, que representa lo bien que la plantilla localiza cada posición. Las plantillas para cada parte son generadas por el cálculo de la imagen media de cada rasgo que hayamos etiqueado durante el entrenamiento. Además se calcula en que posiciones relativas deben encontrarse entre sí. Obteniendo finalmente una colección de rasgos dispuestos en una configuración deformable.

Nosotros decidimos utilizar Boosted Cascade Viola and Jones [2001], este trabajo se distingue por tres contribuciones clave. La primera es la introducción de una nueva representación 'Imagen Integral' que permite calcular rápidamente los features utilizados por nuestro detector. El segundo es un algoritmo de aprendizaje, basado en AdaBoost, que selecciona un pequeño número de características visuales críticas de un conjunto más grande y produce clasificadores extremadamente eficientes Freund and Schapire [1995]. La tercera contribución es un método pa-

ra combinar clasificadores cada vez más complejos en una 'cascada' que permite que las regiones de fondo de la imagen sean descartadas rápidamente mientras se gasta más cálculo en regiones prometedoras de tipo objeto. Además viene implementado en el paquete de procesamiento de imágenes de Matlab o en OpenCV, es un paper sumamente citado en diversos trabajos y es la técnica más utilizada actualmente para la identifiación de rasgos faciales.
De todas maneras, pese a la popularidad del trabajo, corroboramos su rendimiento sobre las fotos de la base de Siblings y obtuvimos la siguiente efectividad en la detección de cada rasgo:

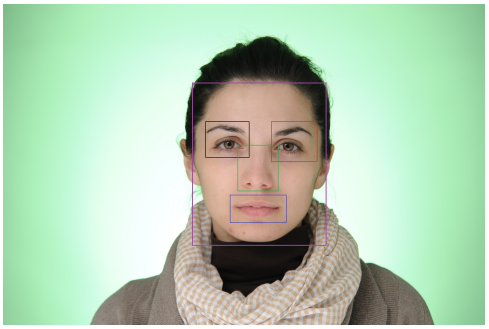

Fig. 2.2: Rasgos a detectar.

Rasgo	Efectividad en detección
Cara	99 %
Boca	66 %
Nariz	94 %
Ojo derecho	82 %
Ojo izquierdo	84 %

Podemos observar que para la cara y la nariz el rendimiento es casi perfecto, para los ojos muy bueno y para la boca bastante aceptable. Por esto es que decidimos proseguir con nuestro trabajo utilizando este algortitmo de extracción de rasgos.

Si bien los resultados de esta extracción automática son muy aceptables, nos interesa mejorar aún más estos rendimientos. Para ellos desarrollamos un script para señalar a mano las coordenadas y de esta manera proveer información supervisada por humanos que aumente la precisión de extracción de atributos. Esto lo hicimos para cada foto de cada una de las bases excepto para la de *Kinface* ya que esta venía con un archivo donde contenía estas coordenadas. La intención es generar los modelos con ambos métodos y comparar sus efectividades, en cuanto a identificación de parentescos, para determinar así si lo logrado a través de la extracción automática es similar a una extracción mas detallada o si hay que seguir perfeccionando este proceso.

Es importante también describir como calculamos los features relacionados a los colores y distancias. Para esto determinamos las coordenadas de la foto en donde se encuentran la nariz, la boca y los ojos y el área que ocupan alrededor de este punto (mediante Boosted Cascade o a mano como mencionamos anteirormente). Luego calculamos el color promedio de esas áreas y tomamos los respectivos valores de cada gama RGB. En el caso del color de piel tomamos el color promedio de la nariz y para la escala de grises previamente transformamos la gama de RGB mediante la función de matlab *rgb2gray*. Para obtener las distancias lo que hicimos fue tomar los puntos centrales del área de cada rasgo y calcular distancia entre los mismos mediante el calculo de distancia en un plano cartesiano:

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Para obtener los tamaños calculamos el área de cada rasgo y lo dividimos por el área de referencia, por ejemplo:

$$\frac{areaNariz}{areaCara}$$

Además con el objetivo de facilitarle la construcción del modelo al machine learning, reducir las dimensiones de la matriz (ya que sabemos que esto mejora el rendimiento del algoritmoBishop [2006]), homogeneizar el funcionamiento de los distintos algoritmos (es decir que tengan buen funcionamiento en todas las bases), aplicamos distintas operaciones a los features extraídos.
Dado que el feature color (de ojo,piel y boca) lo descomponemos en 3 gamas (Rojo, Verde y Azul), en vez de poner el valor de cada una para los dos sujetos lo que hicimos fue entenderlos como valores de un vector de 3x1 y calcular la norma de la resta entre ambos:

$$norma([rojoSujeto1, verdeSujeto1, azulSujeto1] - [rojoSujeto2, verdeSujeto2, azulSujeto2])$$

y la diferencia eucledіana:

$$[rojoSujeto1, verdeSujeto1, azulSujeto1] * [rojoSujeto2, verdeSujeto2, azulSujeto2]$$

Ambas operaciones nos devuelven un único valor correlacionado con los rasgos de ambos individuos. Al reemplazar estos valores en todos los rasgos compuestos por colores, es decir color de ojos, piel y boca, pasamos de tener 18 valores, 6 por cada par de rasgos, a solamente 6, 2 por cada par de rasgos. Lo que reduce considerablemente la dimensión de la matriz, abonando a la generación de un mejor modelo.

2.3. Construcción de la matriz para machine learnig

En una primera instancia generamos una matriz de sujetos x features. Este proceso consta de iterar foto por foto, detectar las coordenadas de los features y calcular los valores antes mencionados. Luego utilizamos esa matriz para generar la final. Para esto formamos todas las combinaciones entre las filas de esta matriz (es decir, entre sujetos), calculando en cada caso las combinaciones entre los rasgos que mencionamos antes (las normas y diferencias euclidianas) y agregándole una columna binaria más que indica si tienen relación de consanguinidad (1) o no (0). Finalmente, obtenemos una matriz con las siguientes columnas, donde todas las celdas tienen valores numéricos y la última es binaria:

- Skin color norm
- skin color euclidean
- right eye color norm
- right eye color euclidean
- left eye color norm
- left eye color euclidean
- distance eye

- distance eye nose
- skin gray color
- distance eye 2
- distance eye nose 2
- skin gray 2
- mouth color norm
- mouth color euclidean
- kin

Por otro lado, con el objetivo de seguir reduciendo la dimensión de la matriz de pares de sujetos, para ver si esto mejora el modelo, decidimos aplicarle análisis de componentes principales antes de usar los distintos algoritmos de clasificación. Esta es una forma de identificar patrones en los datos y expresarlos de tal manera que resalten sus similitudes y diferencias. Dado que los patrones en los datos pueden ser difíciles de encontrar en conteztos de alta dimensión, donde la representación gráfica no está disponible, PCA es una poderosa herramienta para analizar los datos. Otra ventaja de PCA es que una vez que haya encontrado estos patrones en los datos, se comprimen sin mucha pérdida de información.

El análisis de componentes principales o ACP (PCA en inglés) se utiliza mucho en el análisis exploratorio de datos y para construir modelos predictivos, su objetivo justamente es reducir las dimensiones de una matriz tomando un conjunto de variables posiblemente relacionadas y convirtiéndolas en un conjunto de valores sin correlación lineal a través la descomposición en autovalores de la matriz de covarianza Bishop [2006].
Establecimos un rango de varianza para probar que va desde 0.55 (no menos, para evitar perder demasiada información) hasta 0.75 y probamos dentro de ellos cual daba mejor.
Vale aclarar que también probamos generar el modelo sin aplicar PCA (mas adelante lo veremos en la parte de resultados 3.2) y que se nos ocurrió esto, cuando vimos que los funcionamientos entre los distintos algoritmos eran disimiles, con la intención de eliminar ruido y homogeneizar los resultados con un mismo clasificador.

2.4. Machine learning

Uno de los métodos utilizados en la inferencia de clases es el aprendizaje automático,rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender. De forma más concreta, se trata de crear programas capaces de generalizar comportamientos y particularmente en el caso del aprendizaje supersivado Cristianini and Shawe-Taylor [2000], a partir de una información suministrada en forma de ejemplos.
Mas precisamente, en esta rama del aprendizaje automático,se utiliza una técnica cuyo objetivo consiste en crear una función h llamada hipótesis o modelo a partir de un conjunto de instancias etiquetadas. Luego, una vez obtenida h, se utilizará para clasificar nuevas instancias válidas del modelo que ésta define.
Los algoritmos supervisados con que se define h intentan extraer aquellas propiedades que permiten discriminar mejor la clase de cada instancia, y como consecuencia requieren de una clasificación previa (supervisión) de las mismas. En general, las instancias que forman el conjunto de aprendizaje se componen por pares del estilo <arreglo de atributos, clase del objeto>. La misión de los algoritmos de aprendizaje supervisados, es por tanto, construir un modelo a

patir de un conjunto de datos conocidos, que permita hacer predicciones sobre nuevos datos desconocidos.
Parte del desafío de este trabajo fue crear modelos de clasificación basados en esta técnica, utilizando como atributos los rasgos faciales y estableciendo como clases si tienen relación de consaguinidad de primer grado o no. Luego mediante los modelos generados inferir en otros casos a que clase pertenecen.
Para este proceso de generación de modelos de predicción utilizamos la herramienta de machine learning *Weka* Hall et al. [2009]. Lo que hacemos es tomar todas las combinaciones de pares de sujetos posibles, calcular los valores de sus rasgos (esto se explica mejor en la sección *Construcción de la matriz para machine learnig*) y cargar toda esa información en el Weka.

2.4.1. Clasificadores

En cuanto a los algoritmos optamos por utilizar el mismo que en el trabajo inicial Fang et al. [2010], vecinos mas cercanos y agrear nuevos con la intención de obtener mejores predicciones, entre ellos hubo varios con los que obtuvimos muy buenos resultados, estos son: Iterative Classifier Optimizer, Logistic, Bayes Net y Random Forest. Es importante entonces que sepamos como funcionan estos clasificadores.
Iterative Classifier Optimizer es un algoritmo que selecciona el mejor número de iteraciones para el clasificador iterarivo LogitBoost. Este es un *boosting algorithm*Friedman et al. [1998] que se deriva de aplicar *logistic regression* al algoritmo AdaBoost.El algoritmo Adaboost desarrollado por Freund et Schapire en Freund and Schapire [1995] es sin duda el algoritmo del tipo *Boosting* más conocido y estudiado. Utiliza por lo general los arboles de decisión como modelos de base, por lo cual forma parte de los métodos de agregación de modelos homogéneos, y fue pensado originalmente para la clasificación binaria.
A la iteración m del algoritmo, siguiendo cierta distribución, se construye un árbol y se observa las previsiones de éste sobre todos los datos de la muestra. A aquellos datos mal clasificados, se les asignan un peso mayor, influyendo de esta manera sobre la distribución de los datos de la etapa m+1 y forzando al predictor correspondiente a tomarlos "más en cuenta". El predictor que se obtiene al final resulta de un voto mayoritario ponderado o de un promedio ponderado de los predictores de las distintas etapas.Al focalizarse sobre los ejemplos mal clasificados, el error empírico disminuye rápidamente. Sin embargo, en vez de contribuir a un sobreaprendizaje sobre los datos utilizados, se prueba que el error de generalización disminuye también, lo cual hace de Adaboost un algoritmo con un muy buen rendimiento.

Otro clasificador que se basa en árboles de decisión para sus predicciones es *Random Forest*Breiman [2001].Los mismos se encuentran formados por un conjunto de reglas lógicas if-then-else, que se crean a partir de los atributos de las instancias del conjunto de aprendizaje. En general, estas reglas son disyunciones y conjunciones basadas en los valores de los atributos de estas instancias, y son almacenadas en sus nodos internos, de modo que para clasificar una nueva instancia, se evalúan las condiciones a medida que se recorren sus nodos hasta alcanzar las hojas donde se encuentran las etiquetas (resultados), las ramas del árbol, por su parte, representan los posibles caminos de acuerdo a las decisiones tomadas en cada paso.
Mas específicamente este algoritmo consiste en crear una serie de árboles de decisión a partir subconjuntos de los datos de entrada seleccionados de forma aleatoria y subconjuntos de las features del modelo también seleccionadas de forma aleatoria, luego con los árboles, se clasificará una nueva instancia dando como resultado la etiqueta más frecuente. Según Breiman [2001], este método es más estable que los árboles de decisión a secas, dado que varía las porciones de la muestra de datos usadas en el aprendizaje, a diferencia de los árboles de decisión, que se basan en una sola partición de ésta, en consecuencia, Random Forest tiene la capacidad de ajustarse adecuadamente a nuevos escenarios desconocidos pero similares entre sí.

Logistic, a diferencia de *Iterative Classifier Optimizer* y *Random Forest*, tiene un enfoque más matemático, el cual consta de ajustar los datos a una función logísticale Cessie and van Houwelingen [1992]. Es decir buscar qué función de este tipo se ajusta mejor a la información ingresada. Hay muchas formas de investigar qué parámetros de entrada son los que mejor funcionan, en este caso (el más común) se utiliza la técnica de cuadrados mínimos.

Siguiendo con los clasificadores que se basan en enfoques matemáticos tenemos *Bayes Net*, el cual utiliza el Teorema de Bayes para llevar a cabo las inferencias. Se fundamenta en la teoría de la probabilidad y combina la potencia del teorema de Bayes con la expresividad semántica

de los grafos dirigidos; las mismas permiten representar un modelo causal por medio de una representación gráfica de las independencias / dependencias entre las variables que forman parte del dominio de aplicación.
Una red bayesiana es un grafo a cíclico dirigido –las uniones entre los nodos tienen definidas una dirección– en el que los nodos representan variables aleatorias y las flechas representan influencias causales; el que un nodo sea padre de otro implica que es causa directa del mismo.
La inferencia en este algoritmo consiste en calcular las probabilidades a posteriori de un conjunto de variables X después de obtener un conjunto de observaciones Y Ben-Gal [2007], basándose en la fórmula del teorema de Bayes:

$$P(X \mid Y) = \frac{P(X \mid Y)\, P(X)}{P(Y)}$$

Por último tenemos al clasificador *Vecinos más cercanos*, el cual, como dijimos al comienzo, es utilizado en Fang et al. [2010]. Este clasificadorDuda et al. [2012] utiliza el criterio de vecindad, este es una de las aproximaciones más conocidas dentro de las técnicas de clasificación supervisada, que exige la definición de una cierta medida de distancia entre los distintos elementos del espacio de representación. Una de las ventajas que presenta ésta técnica de clasificación es su simplicidad conceptual, que quiere decir: la clasificación de un nuevo punto del espacio de representación, se calcula en función de las clases conocidas de los puntos más próximos a él. El algoritmo consta de: una vez entrenado el modelo con casos conocidos, mediante una función de distancia entre el nuevo caso a clasificar y los entrenados, tomar los K (variable a determinar a la hora de correr el clasificador) más cercanos y analizar cual es la clase predominante en ellos. Luego colocar el nuevo punto en esa clase.

2.4.2. Evaluación de los modelos

Para tener una mayor robustez en la validación de nuestros experimentos una de las técnicas que utilizamos fue medir el accuracy(sobre los datos de testeo) variando las porciones de datos de entrenamiento y testeo de los datos de desarrollo mediante cross validation. Lo que se intenta medir con esto es si los resultados de clasificación son independientes de la partición de los datos de entrenamiento y testeo seleccionados para la misma, luego, en caso de ser cierto, podríamos tener indicios de que el modelo es capaz de generalizar la clasificación sobre nuevas instancias. En general, cross validation, se utiliza en entornos donde el objetivo principal es la predicción y se quiere estimar cuan preciso es un modelo que se llevará a cabo en la práctica Devijver and Kittler [1982]. Dentro de los tipos de cross validation disponibles, nos basamos en K-fold cross validation Refaeilzadeh et al. [2009], el cual permite dividir datos en K subconjuntos, tomando uno de los subconjuntos como datos de testeo y el resto (K-1) como datos de entrenamiento, luego de entrenar el modelo, se clasifica el subconjunto apartado como dato de testeo y se almacena el resultado. Este proceso, es repetido durante K iteraciones, con cada uno de las posibles K combinaciones de subconjuntos de datos de entrenamiento y testeo 2.4. Luego, sobre los resultados almacenados, es posible calcular el promedio y la desviación estándar, es decir cuanto varía en módulo entre cada prueba, para sacar conclusiones sobre el modelo estudiado. En la práctica la elección del número K de iteraciones depende del tamaño del conjunto de datos, nosotros decidimo utilizar su valor estándar que es de 10 iteraciones (10-fold cross-validation) Joanneum [2005].

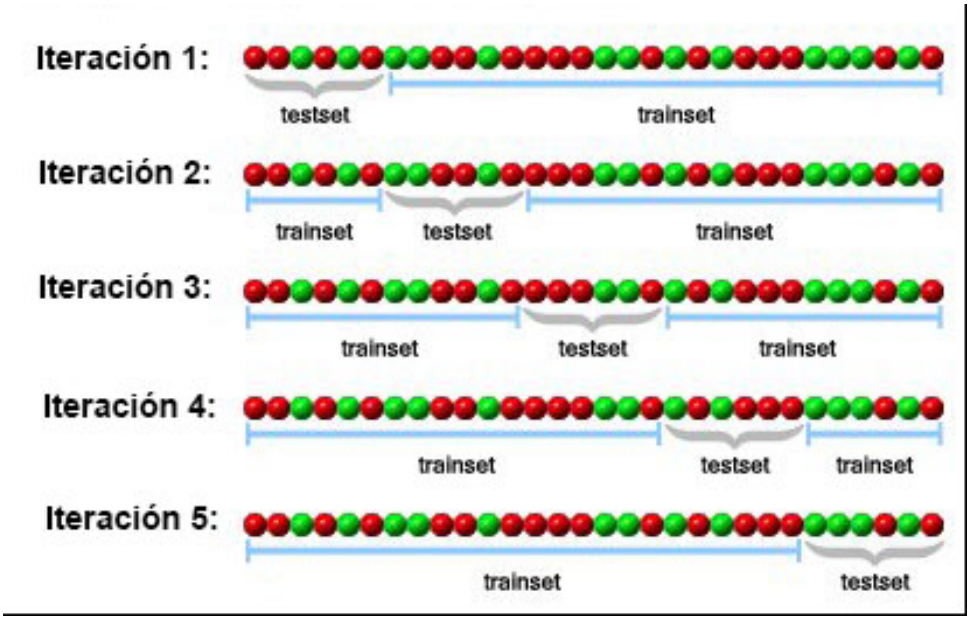

Fig. 2.3: Ejemplo de K-fold cross validation con K=5 iteraciones (https://genome.tugraz.at/proclassify/help/pages/XV.html).

Aún utilizando cross-validation puede surgir un problema a la hora de observar el accuracy, el desbalanceo de clases, es decir que haya muchos más casos positivos (que son parientes) que negativos. En nuestro caso debido a que contamos con 90 pares de hermanos (en el caso de siblings por ejemplo) la cantidad de pares que se puede generar es de 32220 (180x179) osea tenemos 32130 casos negativos y 90 positivos. Este desbalanceo provoca una mala estimación y un mal modelo, para solucionarlo lo que hicimos fue utilizar el filtro de instancias *classBalancer* que nos provee Weka, de esta forma antes de correr los algoritmos equiparamos la cantidad de instancias de cada clase.

Además de variar los testsets y datasets en distintas iteraciones y balancer las clases, es importante tener buenos parámetros de medición al establecer si un modelo de predicción es bueno o malo, es por eso que uno de los datos que analizamos a la hora de medir la efectividad fue el área bajo la curva ROC(Receiver Operating Characteristic)Hanley and McNeil [1982]. Esta curva presenta la sensibilidad o verdaderos positivos (vp) en función de la especificidad o falsos positivos (fp) para distintos puntos de corte. Es decir, cambiando la probabilidad a partir de la cual afirmamos que son padre-hijo nos fijamos qué tasa de vp y fp obtenemos y la marcamos como un punto de la curva ROC. Si la prueba fuera perfecta, es decir, sin solapamiento, la curva pasaría por el punto (0,1) dejando por debajo de ella el cuadrante de ejes positivos y generando así un área igual a 1 bajo ella. Si la prueba fuera inútil: ambas tasas coinciden y la sensibilidad es igual a la proporción de falsos positivos, la curva sería la diagonal de (0,0) a (1,1). Lo común es que las curvas se encuentren en una zona intermedia entre estas dos. A continuación podemos ver un ejemplo:

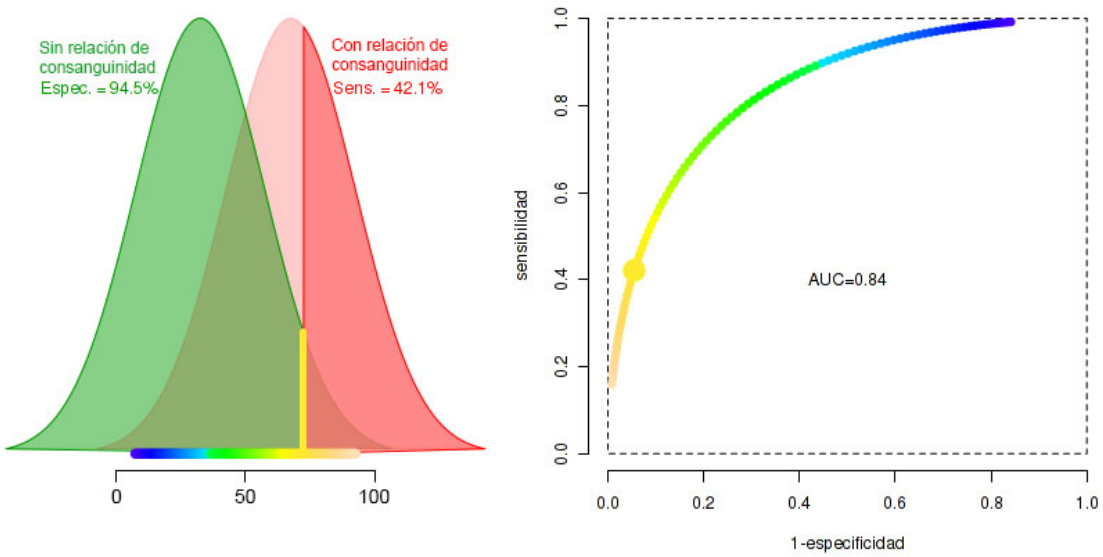

Fig. 2.4: Ejemplo de ROC Curve con un umbral de 70 % (https://www.bioestadistica.uma.es/analisis/roc1/).

Como observamos más arriba, la mayor exactitud diagnóstica de una prueba se traduce en un desplazamiento "hacia arriba y a la izquierda"de la curva ROC. Esto sugiere que el área bajo la curva ROC se puede emplear como un índice conveniente de la exactitud global de la prueba y, en nuestro caso, puede interpretarse como la probabilidad de que ante un par de pares de individuos, unos con relación de consanguinidad y los otros sin, la prueba los clasifique correctamente.

Por último, recordemos que en este trabajo nos basamos en rasgos faciales, identificables y distinguibles en fotos, para poder predecir las relaciones de parentescos entre las personas. Estos rasgos los comparamos entre individuos para, mediante los clasificadores de machine learning, generar modelos que luego identifiquen nuevas relaciones. Nos interesa entonces, poder identificar si la generación de estos modelos varía según el género del padre, ya que el mismo puede inferir directamente en los rasgos de su hijo. Es decir puede haber rasgos que tiendan a ser mas herditables por parte de la madre por ejemplo, lo que nos provocaría que un modelo generado por el sexo femenino del progentior tienda a ser mas eficiente.

En el paper Fang et al. [2010] también generan modelos separando por el género del padre y del hijo, es decir 4 subconjuntos. Esto les da que con madres funciona mejor que con padres, mas precisamente: Padre-hijo 72.9 %, Padre-hija 54.55 %, Madre-hijo 73.81 % y madre-hija 61.29 % . Con esto en cuenta y lo publicado en Jobling and Tyler-Smith [1995] es que decidimos hacer una

separación parecida pero sólo por el género del padre, osea Padre-hijo/a y Madre-hijo/a, para corroborar si las madres tienen una mejor correlación de rasgos con sus hijos y si esa mejora esta dentro del rango del incesto, lo cual podría indicarnos que no es una cuestión genética sino mas bien social. La separación la hicimos manualmente, viendo nosotros mismos foto por foto.

2.5. Identificación de relaciones positivas no reconocidas

Así como es importante saber qué rasgos, qué tipo de normalización de imágénes o qué género del progenitor nos ayudan a generar un buen modelo, es importante identificar también que condiciones son desfavorables. Para esto utilizamos un procedimiento de ingeniería inversa kn2 [2003], el cual consta de, una vez generado el modelo, identificar qué falsos negativos genera y en base a ellos y a los resultados obtenidos con las disitntas pruebas de normalización, extracción de rasgos y género, poder inducir qué características hacen errar a nuestro modelo.

Esto lo chequeamos en la base *Kinship* ya que es la más normalizada en relación a *Kinface* y tampoco tenemos en cuenta *Siblings* ya que las relaciones son de hermanos. Por otro lado, utilizamos el clasificador *Iterative Classifier* ya que es el que mejor anduvo en dicho dataset, arrojando porcentajes por encima de la capacidad humana y de lo logrado en Fang et al. [2010]. Por último, para identificar que relaciones no fueron reconocidas utilizamos la herramienta de predicción que nos provee Weka, donde para cada una de las instancias nos dice cual fue la predicción del modelo.

3. RESULTADOS

En esta sección expondremos los resultados obtenidos a través del proceso experimental previamente explicado y las métricas elegidas. Es decir qué porcentajes de aciertos y Roc Curve obtuvieron los modelos generados por los algoritmos en cada una de las bases mediante los distintos métodos de extracción.

3.1. Selección de los K en Vecinos Más Cercanos

Una de las variables fundamentales a la hora de utilizar el clasificador Vecinos más cercanos es el K que determinemos. En el trabajo del cual partimos Fang et al. [2010] utilizan un K=11, nosotros decidimos probar otras variantes de este valor para analizar cual funciona mejor, en cada base de imágenes en particular, dado que al tener distintas normalizaciones sus dispersiones pueden ser distintas y en todas en general, es decir si hay algún K que funcione de manera aceptable transversalmente a todas las bases.
En el siguiente gráfico3.1 podemos observar la tasa de aciertos obtenida para las 3 bases con distintos valores de K. Como vemos para la base Siblings todos los resultados se encuentran dentro del área de aleatoriedad, con ningún valor se logró una buena performance. Distinto fue Kinface, donde se lograron buenos resultados para los K igual a 11, 25 y 50 con una tasa de 91.97 %, 88.41 % y 82.19 %, respectivamente con unas desviaciones estándar de 4.44 %, 4.29 % y 4.13 %. Para Kinship se obtuvieron resultados aceptables pero no con los mismos valores, con K igual a 150, 200, 250 y 300 las tasas de aciertos fueron de 67.11 %, 64.24 %, 61.85 % y 64.71 % con unas desviaciones de 8.99 %, 8.86 %, 7.82 % y 8.18 % respectivamente. Mientras que para el resto de los valores su perfomance fue aleatoria.

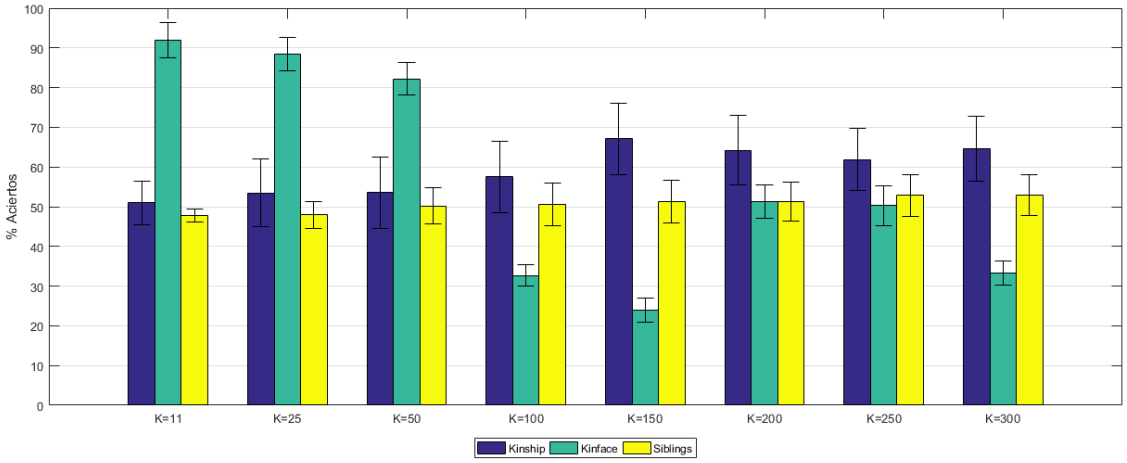

Fig. 3.1: Resultados de vecinos mas cercanos para distintos valores de K

Como no logramos obtener buenos resultados para un mismo K decidimos utilizar dos valores del mismo, uno es K=11 que además de ser el utilizado en Fang et al. [2010] es el que mejor funcionó en Kinface y K=150 que es el que mejor resultado obtuvo en Kinship.

3.2. Clasificación por algoritmo

Como desarrollamos en la sección anterior una de las primeras pruebas que hicimos fue chequear el funcionamiento de los algoritmos de machine learning en las distintas bases de imágenes. Esta prueba la hicimos tanto para la extracción automática de rasgos como para la manual. A continuación podemos ver el desempeño medido en el porcentaje de aciertos y el ROC Curve. Los valores están agrupados por algoritmo.En los gráficos de aciertos 3.2 3.4 3.12 3.14 3.22 3.20 3.6 3.8 3.10 además hay dos lineas de referencia marcadas. Una roja en el 50 % y otra verde en el 67.19 % , la primera representa la aleatoriedad, donde cualquier valor que este cerca de él es totalmente descartable. La verde representa la capacidad humana para establecer este reconocimiento según lo investigado en Fang et al. [2010].
A continuación veremos cuales fueron los valores obtenidos para cada algoritmo bajo los distintos métodos de extracción de rasgos en cada una de las bases de imágenes.

Bajo la modalidad de extracción manual en la base *Siblings* podemos observar que obtuvimos resultados excelentes tanto en efectividad3.2 3.6 como en el ROC Curve3.3 3.7, 95.05 % y 0,95 respectivamente, con el calsificador *Bayes Net*. Además la desviación estándar fue baja, más precisamente de 4.31. Por otro lado también se obtuvieron resultados aceptables en los casos de *Iterative Classifier* (68.65 %) y *Random Forest* (76.99 %), ambos por encima de la capacidad humana y con bajas desviaciones: 4.98 y 5.41. Es interesante observar que acá el área bajo la curva del ROC fue mucho mejor que los aciertos en el caso de *Random Forest* obteniendo un 0.96 con una desviación de 0.04.En cambio en el caso de *Iterative Classifier* el ROC fue igualmente bueno con valor de 0.76.El resto de los clasificadores dieron resultados prácticamente descartables, *Vecinos mas cercanos* con K = 11 Y 150 en la línea del 50 % y *Logistic* un poco por mejor (63.07 %) pero bastante por debajo de la destreza humana.

Al extraer los rasgos de forma automática en la base los resultados obtenidos casi no variaron con respecto a la extracción manual. Es así que detectando automáticamente también generamos un modelo muy bueno con*Bayes Net*, apenas por debajo de la performance manual, con un porcentaje de acierto de 94.95 %3.4 3.6, una desviación de 4.31 y un ROC de 0.95 3.24 con una desviación de 0.04.

Lo mismo se repite con *Iterative Classifier* pero en este caso un poquito por encima (en vez de por debajo) con un 69.16 % de aciertos y un 0.77 en su ROC y *Random Forest* un poco por debajo respecto a la extracción manual con un 74.05 % de acierto, una desviación de 4.97 y un ROC de 0.96 con una desviación de 0.05, sucediendo la misma disparidad entre el ROC y los aciertos.

Al igual que con el otro método de extracción el resto de los clasificadores generaron modelos aleatorios o con un porcentaje de aciertos bastante por debajo del de una persona.

Mediante la extracción manual es aún mas destacable es lo obtenido en *Kinface*, donde funcionaron muy bien los clasificadores *Bayes Net* y *RandomForest*, con porcentajes de aciertos de 94.95 % 3.2 3.8 y un ROC de 0.95 3.3 3.9 en ambos casos. También se obtuvieron resultados muy aceptables, 91.97 % y 0.89, con *Vecinos mas cercanos* con un K igual a 11 y con una desviación de 4.44. Cuando utilizamos un K = 150 la tasa de aciertos fue sumamente baja, 23.98 % con una desviación del 3.06 % y un ROC de 0.33. Al ser tanto más baja que el 50 % deja de ser necesariamente mala, ya que si invertimos sus predicciones obtenemos en una tasa alta los aciertos, en este caso por ejemplo del 76.02 %.
En cambio los modelos generados con *Iterative Classifier* y *Logistic* no fueron buenos, rondando el 50 % lo cual sí es aleatorio.

Distintos fueron los resultados para la extracción automática en esta base, donde no se pudo generar ningún buen modelo mediante la extracción automática de features. El que mejor efectividad dio fue *Logistic* con 53 % 3.4 3.8 de aciertos y 0.56 en su ROC 3.24 3.9, lo cual esta

en la linea de la aleatoriedad y no nos sirve. Todos los demás estuvieron en un 50 % o incluso por debajo.

En *Kinship*, extrayendo manualmente los rasgos, podemos observar que si bien no se generaron modelos tan buenos como en las otras, obtuvimos buenos resultados con *Iterative Classifier* y *Logistic* donde la efectividad 3.2 3.10 y ROC 3.3 3.11 fueron de 71 %|0.72 y 67.42 %|0,76 respectivamente con desviaciones de 9.55 y 9.73.
Vecinos mas cercanos con K = 150 nos dió también una buena tasa de aciertos con un 67.11 % y 8.99 % de desviación, mientras que su área bajo la curva del ROC fue de 0.69. Con el clasificador *RandomForest* podemos ver que se repite lo sucedido en Siblings, con un porcentaje de aciertos malo (50 %) pero un ROC mucho más aceptable (0.65).
Por último los algoritmos *Bayes Net* y *Vecinos mas cercanos* con K = 11 los modelos generados son aleatorios y no nos sirven.

Por último mediante la extracción automática hubo un funcionamiento similar en la mayoria de los clasificadores.*Iterative Classifier* y *Logistic* tuvieron resultados aceptables, con una tasa de aciertos de 71.25 % 3.4 3.10, una desviación de 9.55 y 67.42 % con una desviación de 9.73 respectivamente, ambos por encima de la capacidad humana. Sus valores en el área bajo el ROC curve también fueron buenos: 0.72 para *Iterative Classifier* 0.76 en *Logistic* 3.24 3.11.
El resto de los algoritmos dieron modelos que resuelven la decisión de parentesco de forma random lo cual no nos sirve para nuestro trabajo.

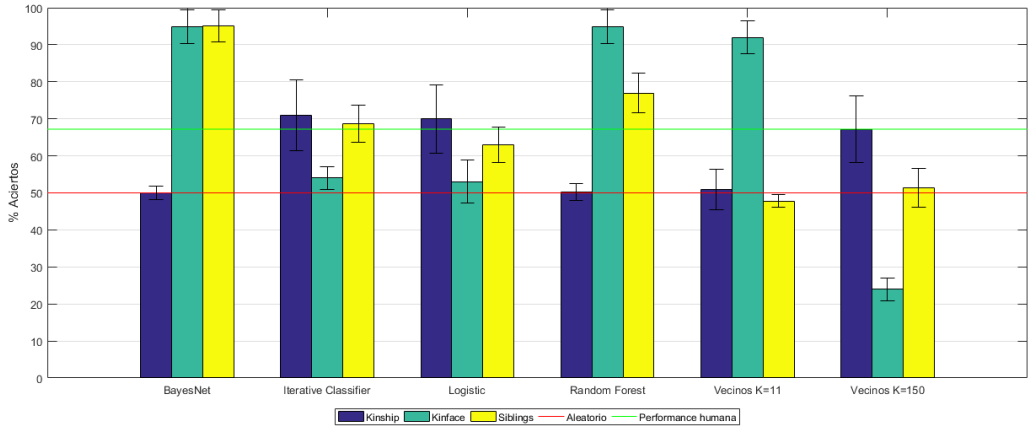

Fig. 3.2: Resultados con marcas manules por porcentaje de aciertos.

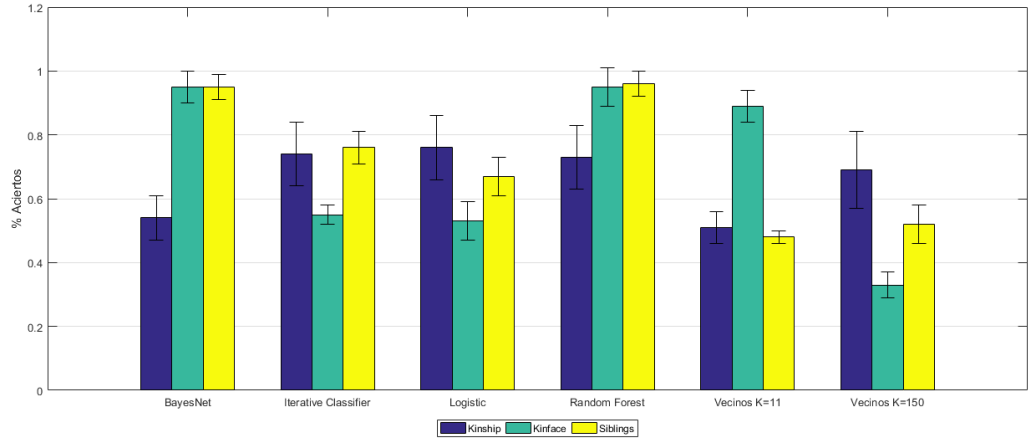

Fig. 3.3: Resultados con marcas manules por ROC Curve.

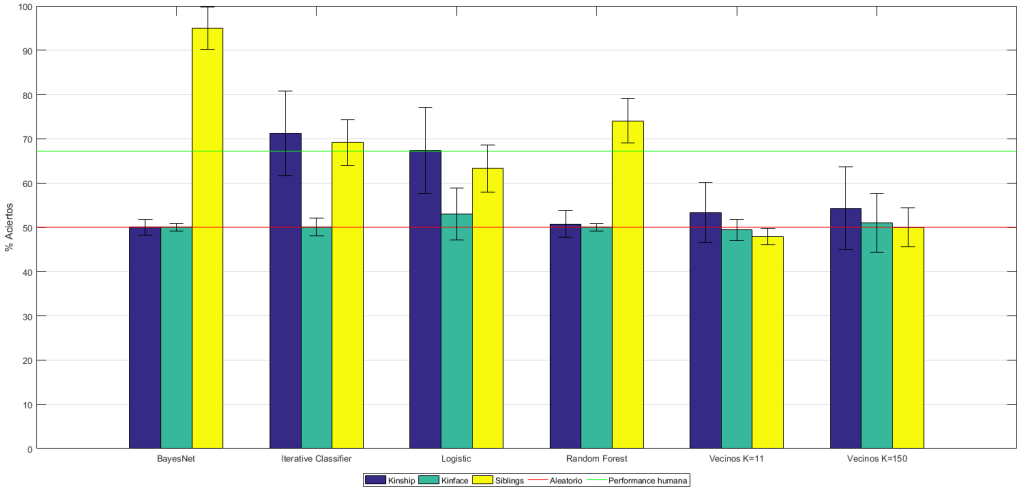

Fig. 3.4: Resultados detección automática por porcentaje de aciertos.

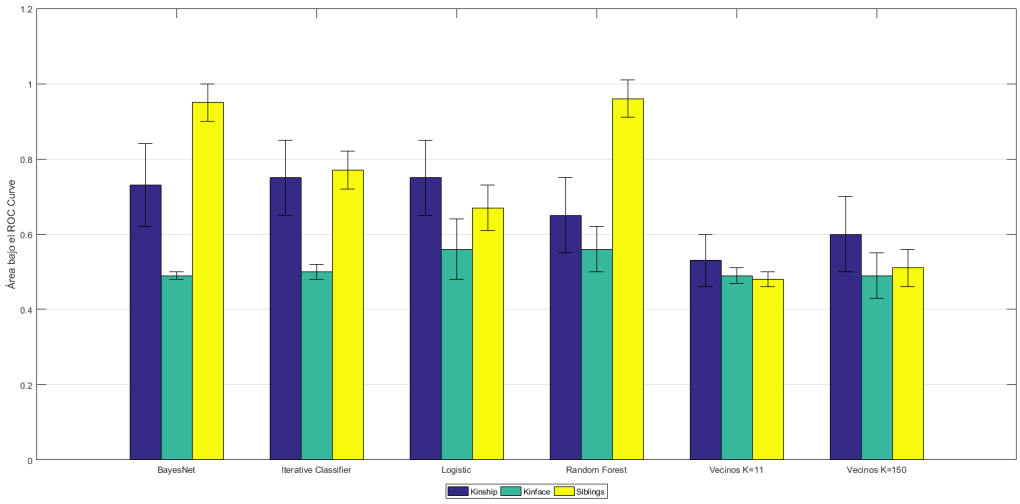

Fig. 3.5: Resultados detección automática por ROC Curve.

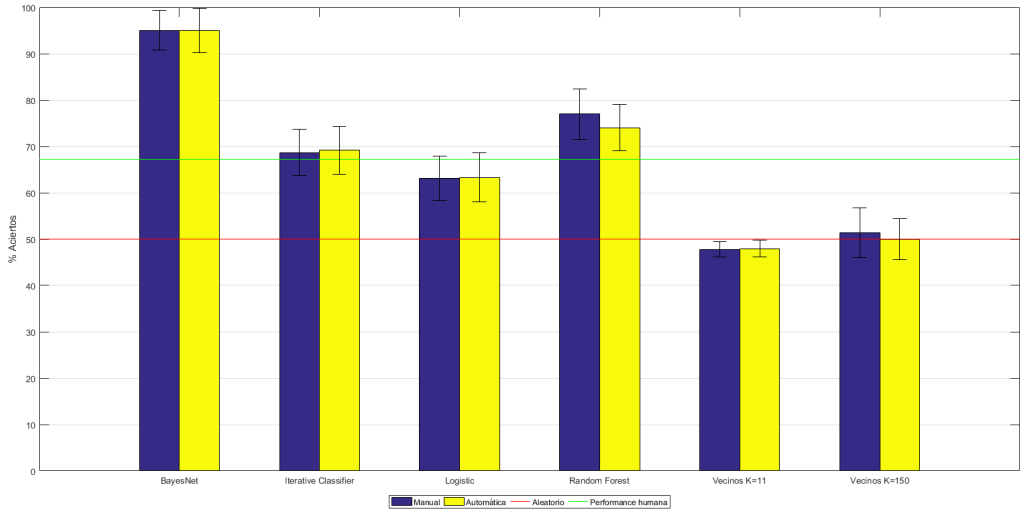

Fig. 3.6: Resultados detección automática vs manual en la base siblings.

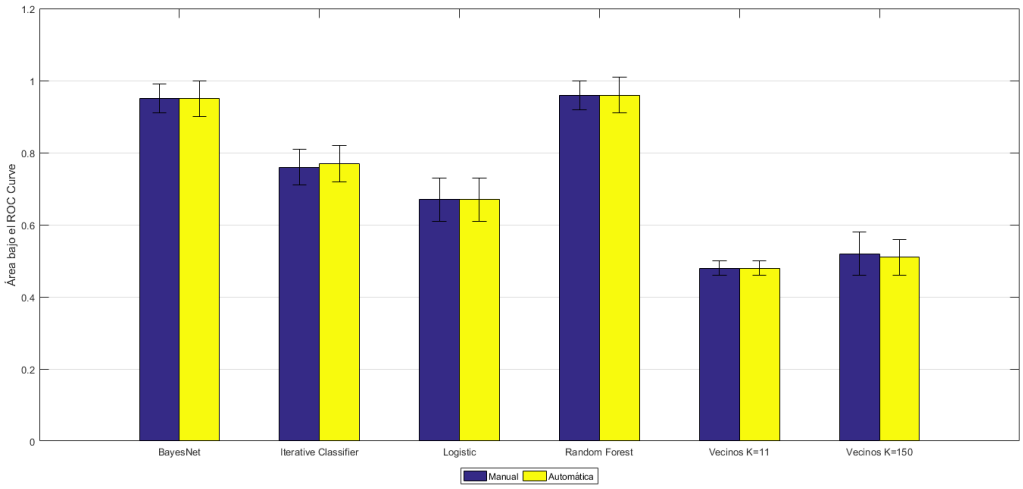

Fig. 3.7: Resultados roc area en detección automática vs manual en la base siblings.

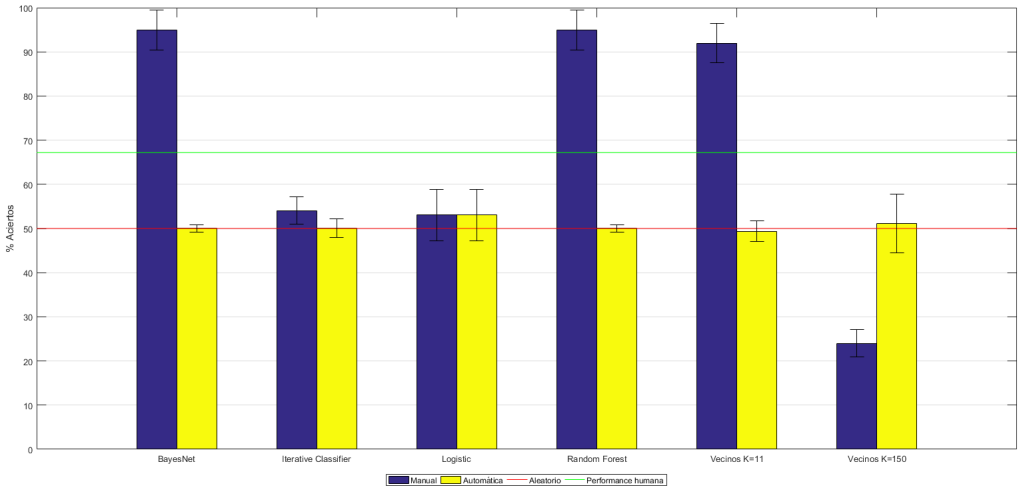

Fig. 3.8: Resultados detección automática vs manual en la base kinface.

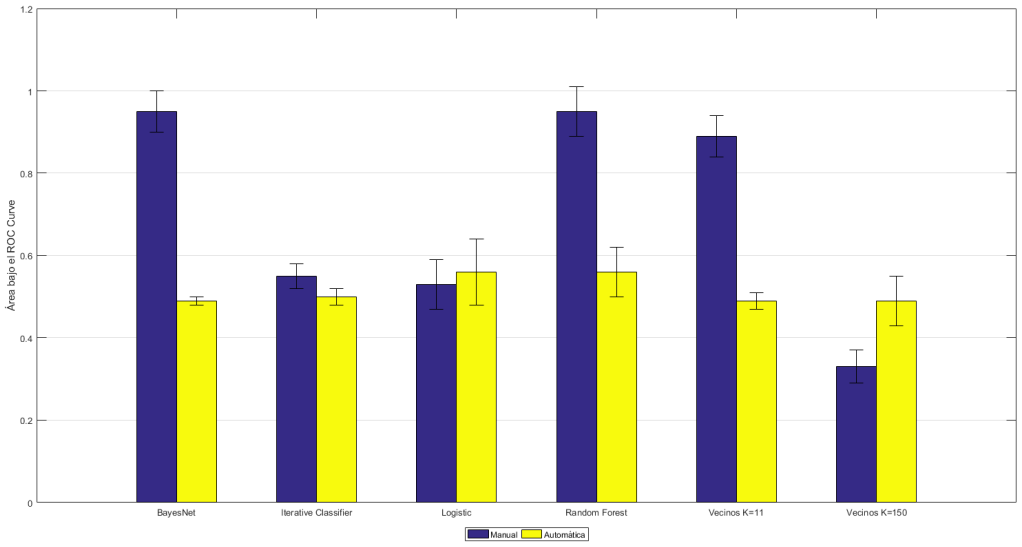

Fig. 3.9: Resultados roc area en detección automática vs manual en la base kinface.

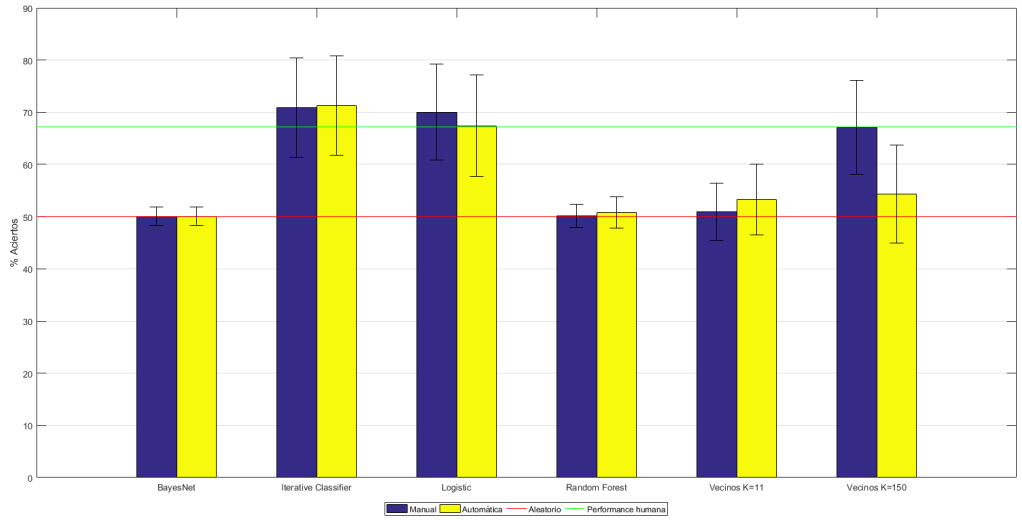

Fig. 3.10: Resultados detección automática vs manual en la base kinship.

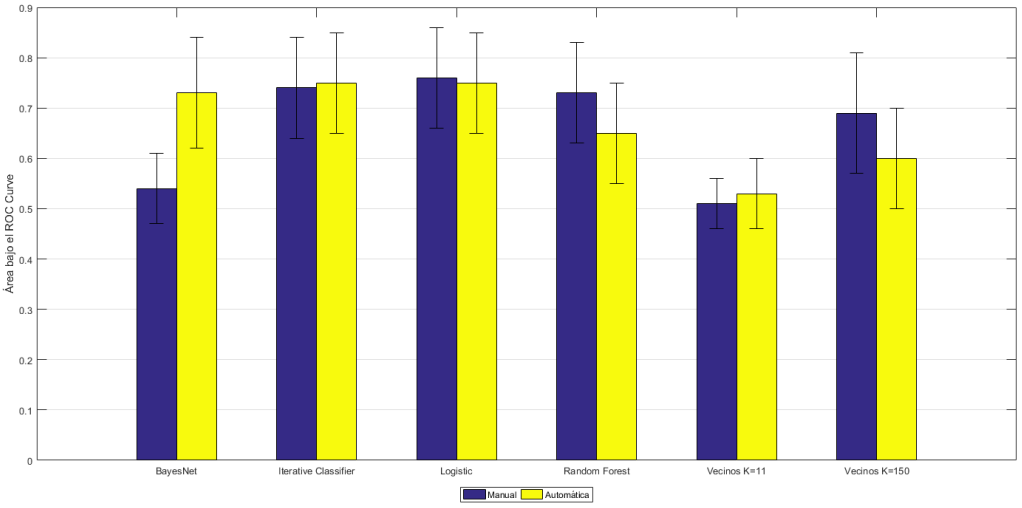

Fig. 3.11: Resultados roc area en detección automática vs manual en la base kinship.

3.3. Clasificación con PCA

Como dijimos anteriormente, con la intención de ver si mediante un pre-procesamiento de los valores de los rasgos mejoran la performance de los modelos generados por los algoritmos de machine learning y para ver si esto nos ayuda a que un mismo clasificador funcione bien en las bases de Kinship y Kinface (las que tienen relación de parentesco, no hermanos), decidimos probar que sucede si a la matriz que le pasamos al clasificador le aplicamos previamente PCA (Principal Component analysis). Las varianzas con las que probamos fueron de 55 % y 75 %. Como el objetivo es ver si esto mejora el modelo lo hicimos sobre las matrices generadas con las marcas a mano, ya que son las mas precisas y nuestro propósito es independiente del tipo de extracción de rasgos que usemos. A continuación observaremos cuales fueron los correspondientes resultados.

3.3.1. 55 % varianza

En estos gráficos podemos observar que *Kinface* tiene excelentes resultados en *Bayes Net*, obteniendo una tasa de aciertos de 95.25 % 3.12 con una desviación de 4.07 y un area bajo el ROC Curve de 0.93 3.13. Aún mejores resultados obtuvimos con *Random Forest* donde podemos ver una muy buena performance de 95.21 % con una desviación estándar de 4.06 y un ROC area de 0.95. *Vecinos más cercanos* con k=11 también obtuvo buenos resultados con un 92.29 % de aciertos, una desviación de 4.00 y un Roc de 0.90 con una desviación de 0.05. El resto de los clasificadores se posicionaron en la línea de la aleatoriedad o por debajo de ella.

En cambio, en *Kinship* los resultados tuvieron distintos desempeños con respecto a *Kinface*, el únicos algoritmo que obtuvo resultados considerablemente por encima de la línea roja (aleatoriedad) fué *Vecinos más cercanos* con k=150 .Con una tasa de aciertos de 67.87 % 3.12, una desviación estándar de 9.62 % y un área bajo el ROC curve de 0.68 con una desviación de 0.123.13. *Iterative Classifier* obtuvo una tasa de aciertos un poco por encima del 50 % pero aún bastante por debajo de la perfomance humana siendo esta del 60.44 % con una desviación del 10.29 %.

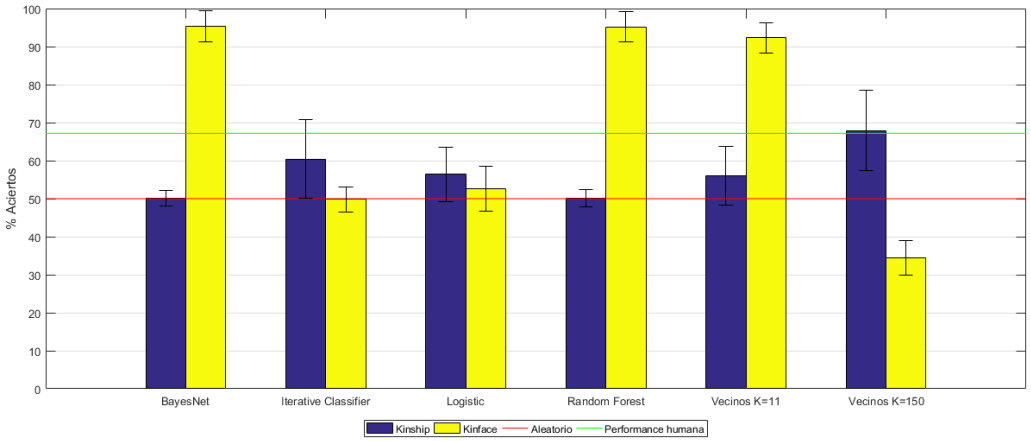

Fig. 3.12: Porcentaje de aciertos habiendo aplicado PCA con una varianza de 55 %.

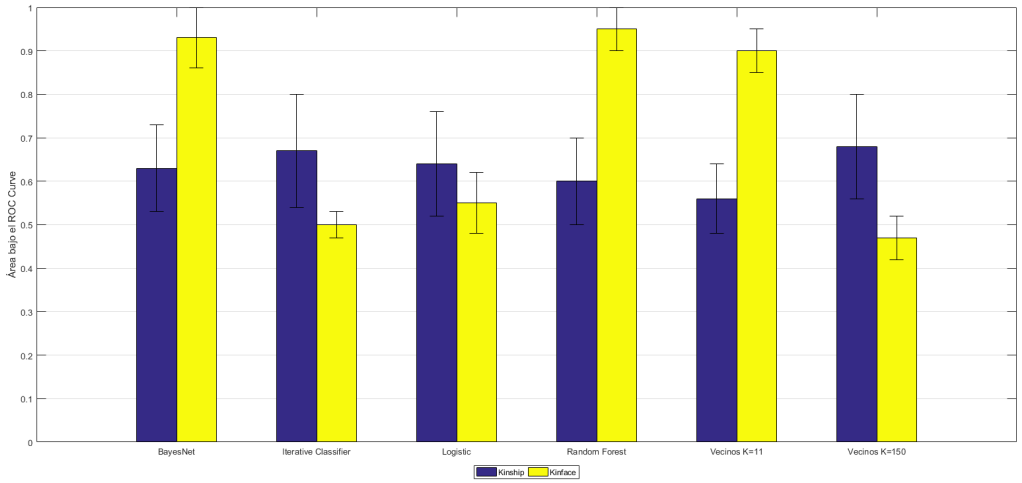

Fig. 3.13: Area bajo el ROC Curve habiendo aplicado PCA con una varianza de 55 %.

3.3.2. 75 % varianza

Al igual que con una varianza de 55 % en el caso de *Kinface* se obtuvieron muy buenos resultados con *Bayes Net* y *Random Forest*. En el primer caso se logró un porcentaje de correctitud del 95.25 %, una desviación de 4.07 % 3.14 y un área bajo la curva ROC del 0.94 con una desviación de 0.07 3.15. En el segundo una tasa de 95.22 % con una desviación del 4.07 % y un ROC de 0.95 con 0.05 de desviación estándar. Por último el otro clasificador que logró una buena performance (los demás estuvieron en la línea de aleatoriedad) fue *Vecinos más cercanos* con k=11 al generar un porcentaje de correctitud del 92.42 % con 4.07 % de desviación estándar y un área bajo el ROC curve de 0.90 con una desviación del 0.06.

Por otro lado en *Kinship* también se repitieron los algoritmos que funcionaron bien pero con una mejoría respecto de la otra varianza en el PCA. En *Vecinos más cercanos* con un k = 150 obtuvimos una tasa de correctitud del 66.70 % con 8.86 % de desviación estándar 3.14 y un área bajo la curva ROC de 0.68 con una desviación del 0.12 3.15.
El resto de los clasificadores también mejoraron en comparación al 55 % de varianza aunque por debajo de lo deseado, siendo *Iterative Classifier* el más destable de ellos con un 62.35 % de correctitud, una desviación del 10.15 % y un área bajo la curva de ROC del 0.66 con 0.12 de desviación.

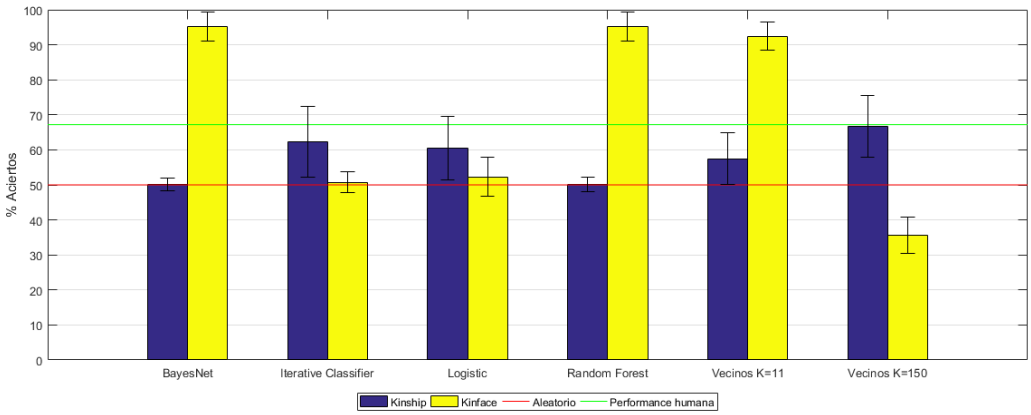

Fig. 3.14: Porcentaje de aciertos habiendo aplicado PCA con una varianza de 75 %.

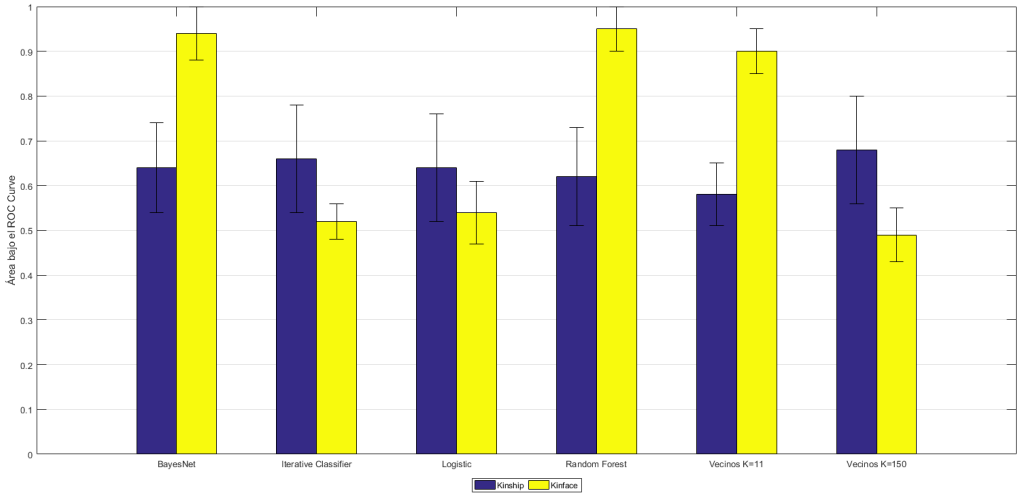

Fig. 3.15: Area bajo el ROC Curve habiendo aplicado PCA con una varianza de 75 %.

3.3.3. Vecinos mas cercanos

Al observar un poco los resultados expuestos en las dos secciones anteriores pudimos notar que con ningún algoritmo logramos obtener buenos resultados en ambos bases. Sin embargo, al mirar un poco mas detalladamente podemos observar que con *Vecinos más cercanos* se obtienen resultados inversos con K=11 y K=150, es decir, en el primer caso funcionó bien en Kinface pero mal en Kinship y en el segundo al revés. Esto nos hizo pensar que con algún valor K intermedio podría lograrse buenos resultados en ambos, por eso es que decidimos también hacer estas pruebas y estos fueron los resultados obtenidos.

55 % varianza

Con esta varianza podemos observar que se lograron muy buenos resultados en Kinface para valores de K iguales a 11,25 y 50, obteniendo tasas de aciertos de 92.24 %, 88.31 % y 82.13 % respectivamente. Todas ellas con desviaciones estándar alrededor del 4 %. Los resultados en sus áreas bajo el ROC también fueron buenas siendo sus valores 0.90, 0.84 y 0.73 con desviaciones del 0.05, 0.04 y 0.04 respectivamente. Con los demás valores de K no se lograron resultados que estén considerablemente por encima de la aleatoriedad.
En Kinship se lograron buenas tasas de aciertos para los K iguales a 150 (como ya vimos) y 300, siendo este último de 67.58 % con una desviación estándar del 9.58 %. Para K igual a 200 el resultado si bien no fue aleatorio, 65.18 %, se encuentra por debajo de la performance humana. En cuanto a los ROC, hubo buenos resultados para K= 100,150,200,250 y 300, siendo estos 0.68, 0.68, 0.71, 0.72 y 0.73 respectivamente, todos con una desviación del 0.12.

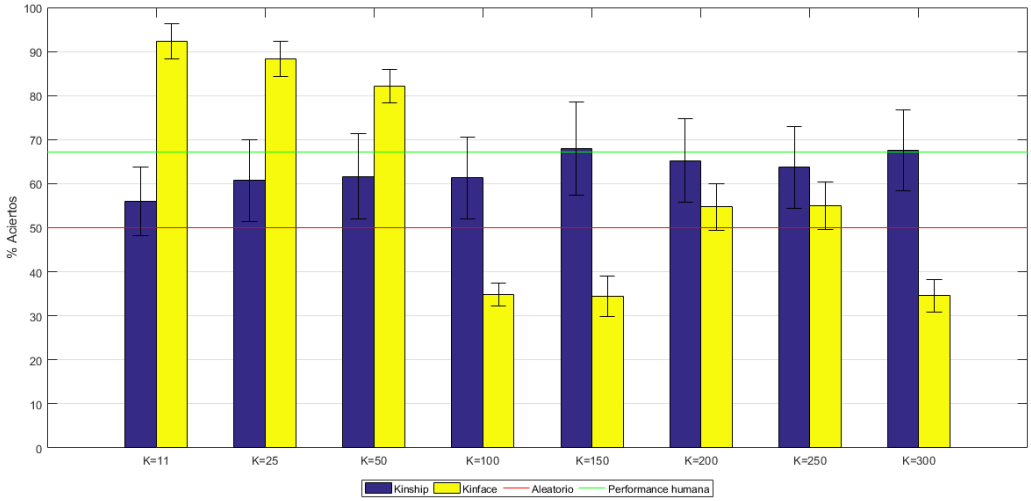

Fig. 3.16: Porcentaje de aciertos habiendo aplicado PCA con una varianza de 55 % para distintos valores de K.

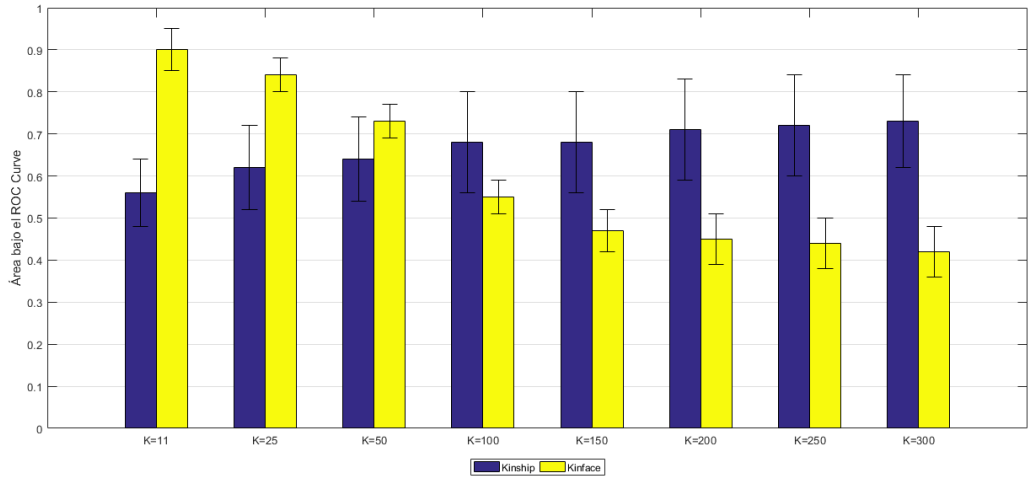

Fig. 3.17: Area bajo el ROC Curve habiendo aplicado PCA con una varianza de 55 % para distintos valores de K.

75 % varianza

Aquí podemos observar que para Kinface se obtuvieron buenos resultados para los mismos valores de K que en la sección anterior (11,25 y 50), siendo ademas las tasas de aciertos muy similares (con diferencias de menos del 0.2 %). Por otro lado sus valores para el área bajo el ROC Curve fueron exactamente iguales.
En Kinship en cambio podemos notar que si hubo diferencias mas grandes con respecto a la varianza de 55 %, siendo para el K = 50 la mas notoria de ellas, donde la tasa de aciertos fue del 66.65 % con una desviación del 8.86 %. Por otro lado también hubo buenas tasas para K=150 (como ya vimos),200 y 250. Donde estos últimos fueron del 67.63 % y 66.47 % con una desviación

estándar del 8.63 % y 8.66 % respectivamente. Para los valores bajo la curva de ROC se obtuvo un 0.69 con una desviación del 0.1 para K=50, también con una importante mejoría respecto a la varianza del 55 %. El resto de los resultados tuvieron una variación menor al 0.01 respecto a la otra varianza.

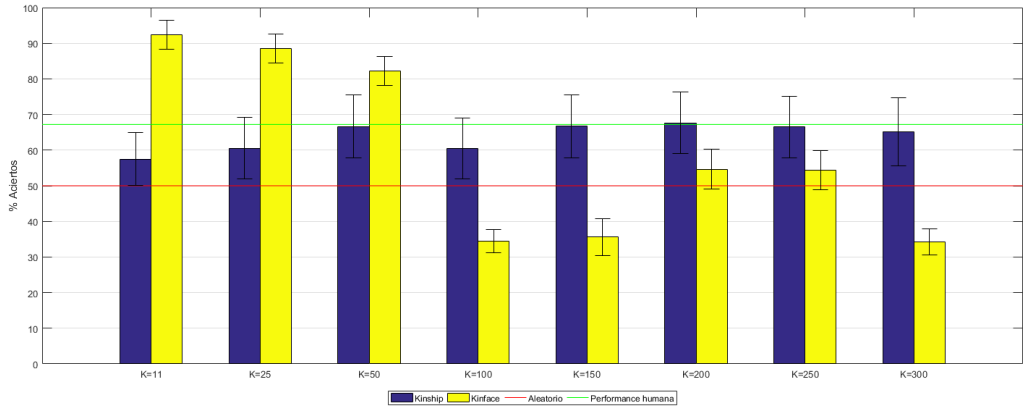

Fig. 3.18: Porcentaje de aciertos habiendo aplicado PCA con una varianza de 75 % para distintos valores de K.

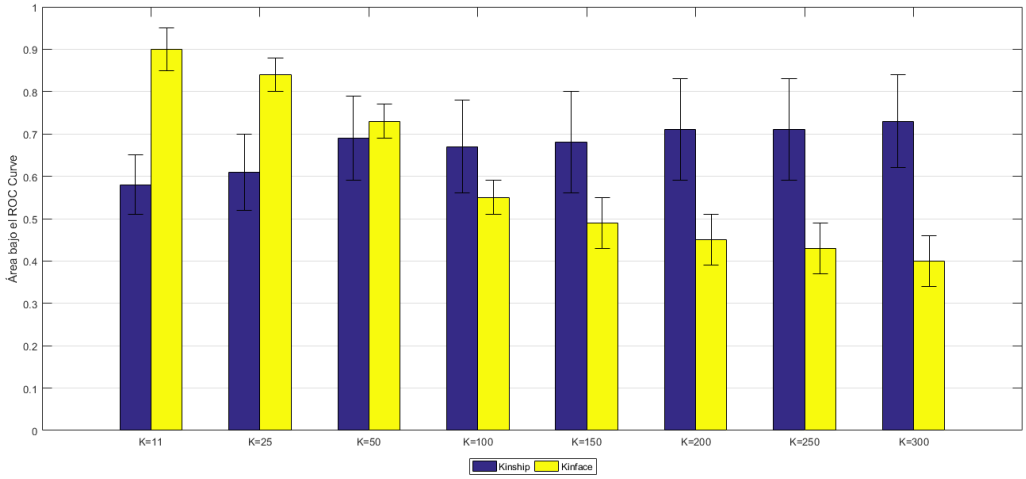

Fig. 3.19: Area bajo el ROC Curve habiendo aplicado PCA con una varianza de 75 % para distintos valores de K.

3.4. Clasificación por género del padre

En esta sección expondremos los resultados que obtuvimos al generar los modelos pero dividendo los datasets en 2, por un lado los hijos con sus madres y por el otro con sus padres. En Kinface la relación es de 13 madres y 87 padres y en Kinship de 44 madres y 99 padres. Partiendo de lo investigado en Jobling and Tyler-Smith [1995] queremos ver si esto tiene alguna incidencia en la generación de los modelos, es decir si los modelos generados sólo con madres funcionan mejor que el de los padres y si esta mejora está dentro del rango de padres ilegítimos expuesto en su investigación. También nos interesa ver si alguna de estas divisiones funciona mejor que la conjunción de ambas. Por último es importante aclarar que al igual que la sección de PCA utilizamos las marcas manos para la extracción de features ya que queremos quitar el ruido de la extracción automática para tener un resultado mas confiable.

3.4.1. Kinface

Para el dataset donde sólo contamos con padres tenemos 3 clasificadores que obtuvieron buenos resultados, estos son: *BayesNet*, *RandomForest* y *Vecinos más cercanos* con K=11. El primero de ellos logró una tasa de aciertos del 95.36 % con 4.28 % de desviación estándar 3.20 y un área bajo la curva de ROC del 0.96 con 0.04 de desviación3.21. El segundo tuvo exactamente los mismos valores en sus porcentajes de aciertos y un ROC de 0.95 con una desviación de 0.05. Por último *Vecinos más cercanos* con K=11 tuvo un porcentaje de correctitud de 92.53 % con 4.21 % de desviación estándar y un ROC de 0.90 con una desviación del 0.04.
El resto de los algoritmos se mantuvieron dentro del área de aleatoriedad.

Por otro lado los resultados en el subconjunto de madres tuvieron desempeños similares o mejores, donde 4 clasificadores funcionaron destacablemente. Estos fueron los mismos que en el caso de los padres mas *Iterative Classifier*. *BayesNet* obtuvo un 95.64 % con 11.90 % de desviación3.20 y 0.96 en su aŕea bajo el ROC con 0.12 de desviación3.21. *Iterative Classifier* generó un modelo con una tasa de aciertos de 81.75 %, una desviación de 15.37 % y un ROC de 0.88 con una desviación de 0.13.En *RandomForest* podemos ver un porcentaje de correctitud de 95.64 % y una desviación de 11.90 %, su área bajo el ROC también fue buena con 0.96 y 0.13 de desviación. Por último *Vecinos más cercanos* con K=11 generó un porcentaje de aciertos de 92.23 % con 11.89 % de desviación y un área bajo la curva de ROC de 0.89 con una desviación de 0.13.
Vecinos más cercanos con K=300 y *Logistic* no generaron buenos modelos, ambos con baja tasa de aciertos.

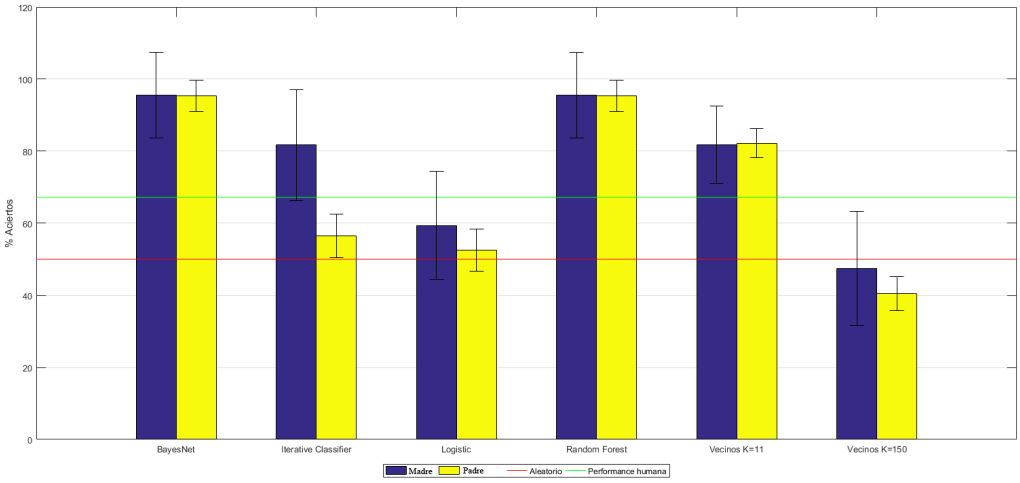

Fig. 3.20: Resultados por porcentaje de aciertos del dataset Kinface por género.

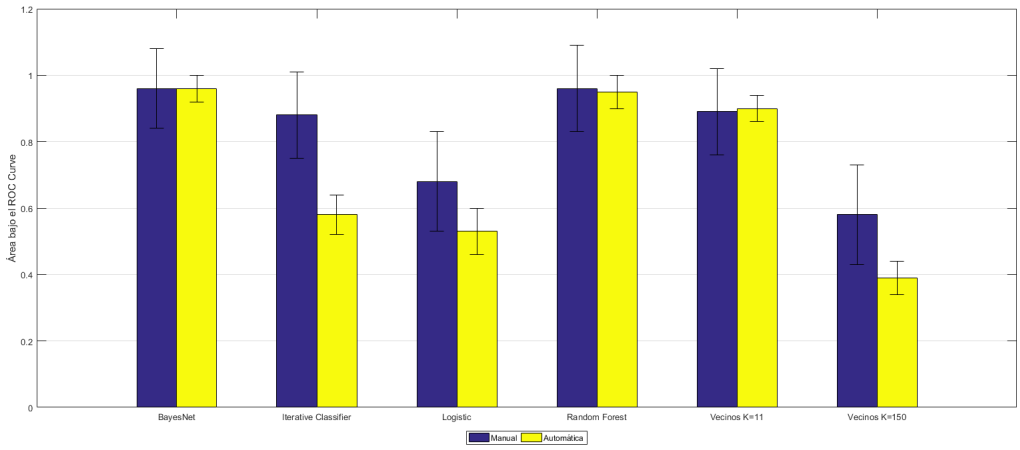

Fig. 3.21: Resultados por ROC Curve del dataset Kinface por género.

3.4.2. Kinship

En la base de imágenes donde hay sólo padres podemos observar que el único resultado que se encuentra de forma considerable por encima de la aleatoriedad es *Logistic* con un porcentaje de aciertos del 63.93 %, una desviación del 13.02 % 3.22 y un área bajo la curva de ROC del 0.7 con 0.14 de desviación estándar 3.23.
El resto de los clasificadores no logra superar la barra del 60 % de aciertos por lo cual no se los toma en cuenta.

En cambio el subconjunto de madres funcionó bastante mejor, el algoritmo *Logistic* logró una performance del 80 % con una desviación del 13.93 % un ROC del 0.88 con 0.12 de desviación.*Iterative Classifier* también tuvo un muy buen desempeño con una tasa del 79.42 %, una desviación estándar del 14.33 % 3.22 y un área bajo el ROC curve del 0.87 con 0.09 de desviación 3.23. Otro que funcionó de forma destacable fue *Vecinos más cercanos* con K=150 donde se obtuvo un 70.02 % de aciertos, una desviación del 14.06 % y un ROC del 0.80 con una desviación del 0.15. Vale notar que todos tuvieron una desviación mas alta de lo normal.
BayesNet y *RandomForest* generaron modelos que predicen esta relación de forma aleatoria y no nos sirven para la investigación en cuestión.

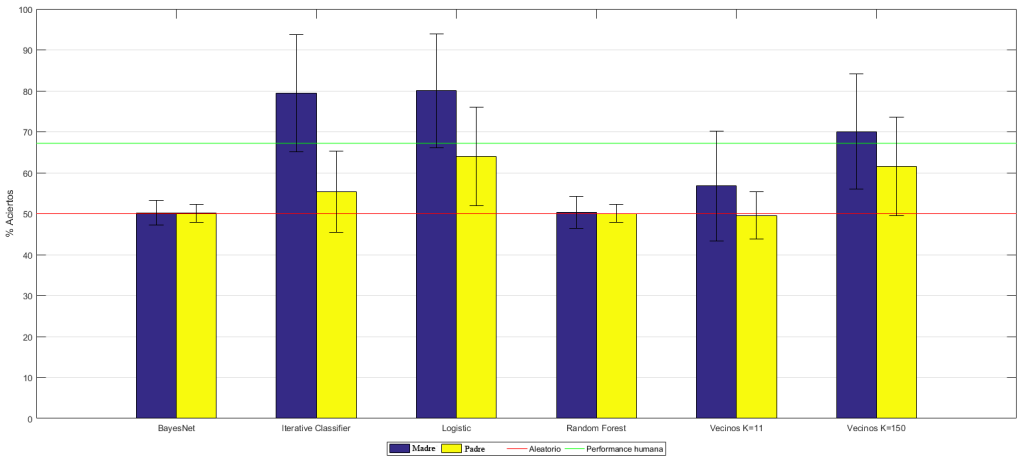

Fig. 3.22: Resultados por porcentaje de aciertos del dataset Kinship por género.

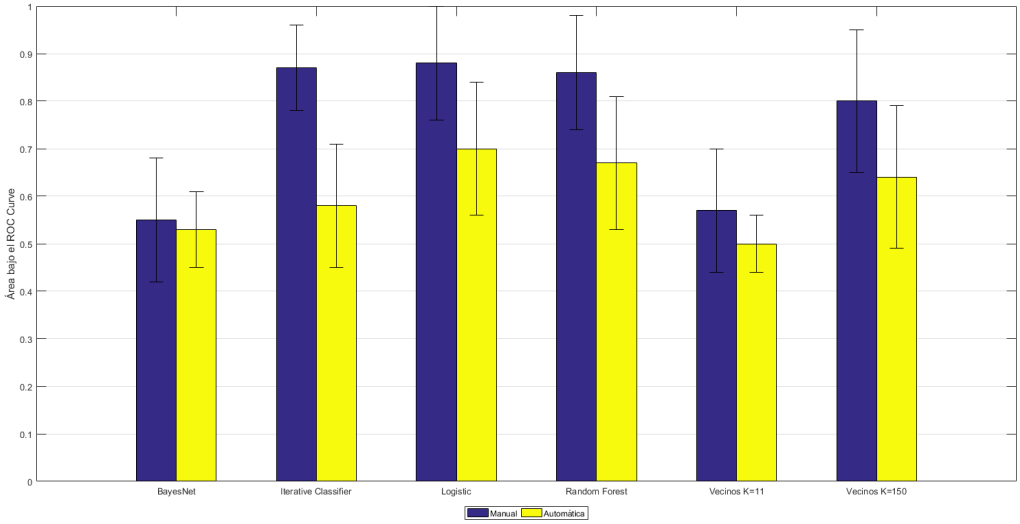

Fig. 3.23: Resultados por ROC Curve del dataset Kinship por género.

3.5. Relaciones no reconocidas

Tras aplicar el proceso descrito en el capítulo Sección 2.5 pudimos identificar cuales son los pares de clases positivas no reconocidas por el modelo generado por el clasificador *Iterative Classifier*.A continuación podemos ver las respectivas imágenes.

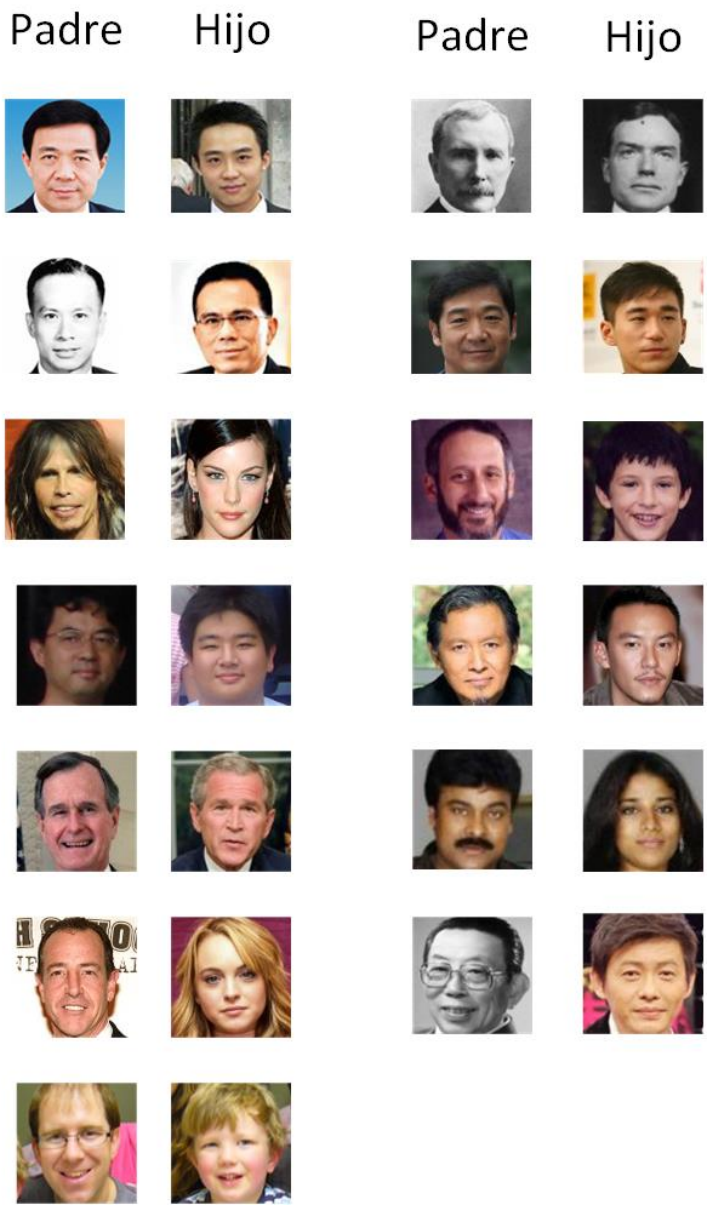

Fig. 3.24: Relaciones de parentesco positivas no reconocidas.

3.6. Predicciones

Una vez generados todos los modelos previamente mencionados decidimos verificar como funcionan en las otras bases de imágenes. Es decir, un modelo generado por un clasificador en la base kinship, por ejemplo, nos fijamos como clasifica en la base kinface y siblings. El objetivo es ver que tan buenos son estos modelos sobre imágenes con otras normalizaciones de las que se utilizaron para crearlos. Para esto tomamos para cada base sus dos mejores modelos y los probamos contra los otros dos datasets. Estos fueron los resultados:

3.6.1. Kinface

Tab. 3.1: Modelos generados sobre la base Kinface

Clasificador	**Base sobre la que se probó**	**% aciertos**	**Area bajo el ROC Curve**
Bayes Net	Kinship	50	0.47
Bayes Net	Siblings	50	0.50
Random Forest	Kinship	50	0.55
Random Forest	Siblings	50	0.50

3.6.2. Kinship

Tab. 3.2: Modelos generados sobre la base Kinship

Clasificador	**Base sobre la que se probó**	**% aciertos**	**Area bajo el ROC Curve**
Logistic	Kinface	52.94	0.50
Logistic	Siblings	53.23	0.47
Iterative Classifier	Kinface	57.88	0.50
Iterative Classifier	Siblings	55.05	0.55

3.6.3. Siblings

Tab. 3.3: Modelos generados sobre la base Siblings

Clasificador	**Base sobre la que se probó**	**% aciertos**	**Area bajo el ROC Curve**
Bayes Net	Kinship	50	0.49
Bayes Net	Kinface	50	0.49
Random Forest	Kinship	50	0.50
Random Forest	Kinface	50	0.52

4. CONCLUSIONES

Tras haber analizado todos los modelos obtenidos en el capítulo anterior hay varios aspectos sobre los cuales podemos reflexionar y llegar a distintas conclusiones.
Como dijimos en la introducción la base de imágenes más normalizada es *Siblings*, la gran mayoría tienen un mismo tamaño, iluminación, definición, expresión y fondo. Si comparamos la performance de los modelos entre la extracción automática y la manual podemos ver que, a diferencia de los otros dos datasets, prácticamente no varían los resultados. Lo cual nos indica que sobre bases de imágenes bien normalizadas utilizar la extracción de rasgos mediante *Boosted Cascade* Viola and Jones [2001] para la generación de un modelo tiene la misma efectividad que establecer a mano las coordenadas de cada rasgo, es decir la performance humana para detectarlos.
Por otro lado podemos ver que en *Kinship* sí hubo algunas variaciones entre las formas de extracción, obteniendo peores rendimientos en los casos automáticos pero no por grandes diferencias. Esto no sucede en *Kinface* donde no hubo un solo clasficiador que tuviese resultados aceptables. Una de las principales diferencias, en cuanto a normalización, entre estas bases es el tamaño de las imágenes. Mientras que en *Kinship* todas tienen las mismas dimensiones (100x100) en Kinface cada foto tiene sus propias medidas. Esto puede ser probablemente un factor importante para la relación extracción de rasgos - generación del modelo.
Es entonces que, con respecto a la extracción de rasgos, podemos concluir que si no tenemos una estricta normalización de las imágenes sigue siendo mejor las marcas manuales, aunque en caso de que sí lo tengamos la eficiencia es muy buena. Como nuestro trabajo tiene la intención de ser utilizado para la detección de los hijos de desaparecidos por la última dictadura cívico-militar esto puede no ser un obstáculo, en el caso de que se pudiese contar con la base de imágenes de los DNI, por ejemplo, donde todas tienen una normalización prácticamente igual a *Siblings*, un mismo fondo, misma iluminación, tamaño y expresión.

Por otro lado, con respecto al funcionamiento de los modelos, logramos mejorar el porcentaje de aciertos obtenido en Fang et al. [2010] no sólo utilizando su misma base de fotos sino que en otras dos bases de imágenes. En todas ellas y con distintos clasificadores obtuvimos porcentajes de aciertos considerablemente por encima de la performance humana llegando, en los casos de *Siblings* y *Kinface*, a resultados cercanos a las pruebas biológicas Lévi-Strauss [1969], Carsten [2000] con un 95.05 %. Además hicimos una segunda constatación de estos valores observando también los números del área bajo la curva de ROC, ya que puede pasar (como efectivamente sucedió) que estos valores difieran en su rendimiento por lo que dejan de ser confiables los resultados obtenidos.
En los casos en que los modelos son buenos sus desviaciones estándar (entre los distintos cross-validations) son aceptables, variando entre un 3 y un 12 por ciento. Una vez más podemos ver que donde menos desviación tuvimos fue en *Siblings*, donde mayor normalización hay.
Para el caso de *Kinface* es importante destacar que cuando probamos con una sola imagen del padre (la de adulto mayor) los modelos no eran buenos, dieron un salto importante en su calidad al agregar las de los padres de jóvenes. Por lo que podemos deducir que tener más imágenes de la misma persona nos ayuda a la generación de un buen modelo de predicción.
Además pudimos chequear que los rasgos que agregamos nosotros respecto a los ya utilizados en Viola and Jones [2001] no hicieron diferencia a la capacidad de predicción de los modelos generados, ya que al sacarlos y volver a generar el modelo los resultados obtenidos eran los mismos. Por lo que deducimos que la mejoría en la performance que logramos se debe a diversas variables. En el caso de Kinship a la metodología de extracción y a la utilización de nuevos

clasificadores ya que es lo que variamos con respecto al trabajo previo Fang et al. [2010]. En las otras dos bases la mejoría se debe además de a estos factores a la mejor calidad y cantidad de imágenes que poseen (como ya explicamos previamente).

Luego, para cada uno de los modelos que generaron buenos resultados, intentamos predecir en otro dataset. Aquí las predicciones no funcionaron, con ninguna de sus combinaciones. Es decir haciendo todas las combinaciones posibles entre clasificador, dataset y dataset sobre el cual predecir (que no sea el mismo sobre el cual se aplico el algoritmo). Esto seguramente se deba a lo que ya mencionamos anteriormente, que entre estos datasets no hay un patrón común de normalización, lo que provoca que los modelos no funcionen entre sí, el modelo se esta creando en función de cierta normalización de sus fotos y no funciona bajo una distinta.

Sin embargo esto también podía estar indicándonos que los buenos resultados podían tener otros motivos o depender de otras variables. Ya que además de que un modelo con buenos resultados en un dataset no funcionaba en otro, no había un mismo clasificador que generase buenos modelos en dos base de imàgenes distintas.Por ejemplo, tanto en *Kinface* como en *Kinship* lográbamos buenos resultados pero no dentro de un mismo algoritmo, los que funcionaban muy bien en *Kinface* no lo hacían en *Kinship* y viceversa.

Por eso decidimos probarlos nuevamente pero tomando sus componentes principales. Esto nos dió que para *Kinface* no hay diferencias significativas entre aplicarlo y no (con ambas varianzas). En cambio si hay diferencia en *Kinship*, aquí la mayoría de los clasificadores tuvieron un rendimiento peor excepto por *Vecinos más cercanos* (el mismo utilizado en Fang et al. [2010]), donde con ambas varianzas creció su efectividad llegando, en el caso de k=50 y una varianza del 75 %, al nivel de la performance humana. Logrando de esta manera que un mismo clasificador obtenga resultados aceptables en ambos datasets. Demostrándonos de esta manera que efectivamente mediante este método estamos seleccionado las variables que realmente nos indican si la relación de consanguinidad de primer grado existe o no.

Yendo a los modelos generados separando por el sexo del progenitor, pudimos apreciar que para el dataset *Kinship* en todos los casos, donde los modelos fueron aceptables, el generado por las madres fue ampliamente más eficiente que el de los padres, incluso mejor que con padres y madres juntos. En la base *Kinface* podemos ver que con los clasificadores en los que había tenido un muy buena perfomance con ambos géneros juntos, por separado tienen la misma performance en cada uno que a la vez es la misma que juntos. En cambio, en el caso de *Iterative Classifier* donde antes se había tenido una performance mala, al separarlos se obtuvo una muy buena en el de madres y una mala en el padres.

Esto lo que nos indica es que los modelos se generan más eficientemente cuando el género del progenitor es femenino. Por otro lado, podemos ver que la diferencia de aciertos en el caso de *Kinface* promedia (entre todos los clasificadores) un 6,69 % y un 9,67 % en el caso de *Kinship*. Esta diferencia está por encima del valor de hijos ilegítimos (4 %) detectado en *Fathers and sons: the Y chromosome and human evolution* pero no por una gran diferencia. Lo que nos indica que esta cuestión sociológica puede estar teniendo un impacto en nuestros modelos. También que los rasgos que seleccionamos (la mayoría de ellos colores: boca, ojos y piel) podrían tener una mayor preponderancia hereditaria en la madre que en el padre.

Por último, observando todos las las relaciones positivas mal predictas podemos notar que en todos los casos el género del progenitor es masculino, teniendo en cuenta lo visto en la sección anterior podemos determinar que esta es la principal variable que afecta negativamente la posibilidad de poder predecir correctamente si o no son parientes.

También podemos decir que hay otras variables en juego que terminan por “confundir“ al modelo, como es el color de las imágenes (osea que algunas sean a color y otras en blanco y negro), la diferencia de edad entre el hijo y el padre, la diferencia de género entre ellos y la diferencia de iluminación en la foto.

5. TRABAJO FUTURO

En este último capítulo desarrollaremos algunas líneas sobre las cuales podría seguir investigándose para profundizar todo lo aquí expuesto, con la intención de poder generar algún día un porcentaje de predicción más cercano al biológico y lo suficientemente universal para no depender del nivel de normalización de las fotos. Para de esta manera poder aprovechar al máximo el poder de cómputo y predicción poniéndolo al servicio de un estado que se preocupe por los derechos humanos y pueda, mediante esta herramienta, aportar a la identificación de posibles candidatos/as sobre los que luego verificar más datos y, en caso de haber datos suficientes, constatar biológicamente .

Para solucionar el problema de la extracción de rasgos en fotos con distintas normalizaciones, o sin normalizar directamente, lo que se podría usar es *Deep Learning* Taigman et al. [2014] . Esto implicaría un trabajo mucho más profundo y de mayor duración, teniendo que entrenar nuestro propio identificador, pero podría lograrse una extracción que sea independiente de los gestos, la posición de la cara, la iluminación, el tamaño, el fondo y todo lo que estuvimos mencionando en este trabajo que representaba un problema a la hora de detectar automáticamente los features faciales.También sería de gran utilidad poder contar con más bases de imágenes de padres/madres e hijos/as para poder hacer una mayor y mejor estadística. Además para nuestro objetivo en particular sería óptimo que esas bases fueran de Argentinos o Sudamericanos. Ya que las morfologías faciales inciden en la generación y predicción de estos modelos. Otro aspecto que sumaría es que poder tener más de una imagen de cada sujeto (la mayor cantidad posible) y, si no se trabaja mediante deep learning, la mayor normalización posible.
Teniendo en cuenta que no es fácil conseguir una buena foto de una persona desaparecida hace mucho tiempo, sería de gran ayuda para la identificación de los hijos de los desaparecidos de la última dictadura cívico militar acontecida en nuestro país (o en cualquier otro) poder extender esta identificación a relaciones de consaguinidad de segundo grado, es decir abuelos/as-nietos. Para de esta manera utilizar mejores fotos, de mayor calidad y mas actuales.

Yendo más hacia el funcionamiento mismo del modelo de predicción, sería de gran utilidad saber en qué casos en los que las relaciones son negativas estos fallan, así como detectamos en que pares donde las relaciones entre los sujetos eran positivas los modelos generaban un falso negativo. Del mismo modo en que en base a la positiva pudimos intuir que características de las fotos podían estar interfiriendo, con esta otra investigación podría detectarse cuales tienden a inducir al modelo a predecir que si son parientes generando falsos positivos.
Además el modelo se podría perfeccionar conociendo cuáles rasgos de la genética biológica son hereditables según el género y contemplar esta probabilidad a la hora de extraerlos, seleccionarlos y pre-procesarlos, teniendo en cuenta cuales son evaluables en una imagen y cuales no.

Por último, sería útil seguir probando con distintas variantes de los parámetros de los clasificadores aquí expuestos y con nuevo algoritmos también. Para ver si así se logra un modelo que se ajuste mejor a la predicción de parentescos de consanguinidad de primer grado mas independientemente de la normalización de las fotos o de la cantidad de ejemplares con que se cuente de un mismo sujeto.

Bibliografía

Claude Lévi-Strauss. *The elementary structures of kinship*. Number 340. Beacon Press, 1969.

Janet Carsten. *Cultures of relatedness: New approaches to the study of kinship*. Cambridge University Press, 2000.

Ana Maria Di Lonardo, Pierre Darlu, Max Baur, Cristian Orrego, and Mary-Claire King. Human genetics and human rights. identifying the families of kidnapped children. *The American journal of forensic medicine and pathology*, 5(4):339–347, 1984.

Ruogu Fang, Kevin D Tang, Noah Snavely, and Tsuhan Chen. Towards computational models of kinship verification. In *2010 IEEE International Conference on Image Processing*, pages 1577–1580. IEEE, 2010.

J. Luo S. Xia, M. Shao and Y. Fu. Understanding kin relationships in a photo. *IEEE Transactions on Multimedia*, 14(4):1046–1056, 2012.

M. Shao S. Xia and Y. Fu. Kinship verification through transfer learning. In *Proc. International Joint Conferences on Artificial Intelligence (IJCAI)*, pages 2539–2544, 2011.

S. Xia M. Shao and Y. Fu. Genealogical face recognition based on ub kinface database. In *Proc. IEEE CVPR Workshop on Biometrics (BIOM)*, 2011.

Tiago F. Vieira, Andrea Bottino, Aldo Laurentini, and Matteo De Simone. Detecting siblings in image pairs. *The Visual Computer*, 30(12):1333–1345, 2014. doi: 10.1007/s00371-013-0884-3. URL `http://dx.doi.org/10.1007/s00371-013-0884-3`.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

Mark A Jobling and Chris Tyler-Smith. Fathers and sons: the y chromosome and human evolution. *Trends in Genetics*, 11(11):449–456, 1995.

Pedro F Felzenszwalb and Daniel P Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

C.M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006. ISBN 9780387310732. URL `https://books.google.co.uk/books?id=kTNoQgAACAAJ`.

Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

Pierre A Devijver and Josef Kittler. *Pattern recognition: A statistical approach*. Prentice hall, 1982.

Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. In *Encyclopedia of database systems*, pages 532–538. Springer, 2009.

FH Joanneum. Cross-validation explained. *Graz, Austria: Institute for Genomics and Bioinformatics, Graz University of Technology*, 2005.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

S. le Cessie and J.C. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201, 1992.

Irad Ben-Gal. Bayesian networks. *Encyclopedia of statistics in quality and reliability*, 2007.

Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification*. John Wiley & Sons, 2012.

James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

Product design: techniques in reverse engineering and new product development. 2003.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.