

Universidad ORT Uruguay

Facultad de Ingeniería

Bernard Wand Polak

Ingeniería en Sistemas

**Machine Learning para Sistemas
inteligentes**

Obligatorio 1

Joaquin Rodríguez (231355)

Matías Olivera(267919)

Santiago Cabrera (266191)

Grupo M6C

Docente: Matías Carrasco

Agustina Disiot

Índice:

Índice:	2
Informe sobre Machine Learning.	3
Introducción:	3
Preprocesamiento:	3
Extracción de features:	3
Dataset:	3
Clasificadores:	4
Árbol de decisión:	4
Regresión Logística:	6
Boosting:	7
Gradient Boosting:	7
ADA Boosting:	9
Modelo 1:	9
Modelo 2:	10
Modelo 3:	12
Random Forest:	14
Modelo 1:	14
Modelo 2:	16
Redes neuronales:	18
Modelo 1:	18
Modelo 2:	20
Elección del modelo:	21

Informe sobre Machine Learning.

Introducción:

Lo primero que hicimos como equipo fue analizar la letra del obligatorio. Lo primero que constatamos es que se trata de un problema de clasificación binaria. De esta manera empezamos a pensar cuales serían los posibles modelos que resuelven este tipos de problemas. Nos enfocaremos en las métricas y en la rapidez de los modelos como parámetros de comparación de los modelos.

Para nuestro trabajo usamos de base las notebooks dadas en clase sin embargo algunas de las decisiones tomadas las explicaremos en este informe. En este informe vamos a detallar los modelos hechos y luego seleccionaremos uno.

Preprocesamiento:

A nivel de preprocesamiento nuestro equipo decidió usar los datos brutos sin ninguna normalización. El único preprocesamiento utilizado para las fotos fue pasarlas al blanco y negro las fotos correspondientes a los fondos.

Extracción de features:

Para la extracción de las features utilizamos las HOG features vistas en clase. Utilizamos distintas escalas para los parches. Nuestra idea además de tener un modelo el cual sea muy bueno a nivel de métricas sino que también sean rápidos de correr es por eso que elegimos bajar un poco la calidad de las features para que los modelos no tengan tanto tiempo de procesamiento.

Utilizamos 3 orientaciones en lugar de 9 por ejemplo para poder correr los modelos más rápidamente. Además de utilizar 12*12 en los pixeles por celda y 3*3 en celdas por bloque.

Dataset:

Para el Dataset después de una pequeña prueba decidimos usar 1 a 10 de proporción para la cantidad de caras respecto a los fondos como para poder asemejarse a la vida real. Aunque somos conscientes que una proporción más parecidas sería 1 a 50.

Clasificadores:

Una decisión para todos los modelos fue usar un Cross-Validation (usamos una de 5 folds ya que es lo que se recomienda) para la evaluación de los modelos para así podremos evaluar mejor su rendimiento al reducir la variabilidad en la evaluación y evitar el sobreajuste en algún caso. Para evaluar cada modelo consideraremos su métricas particulares, como equipo decidimos centrarnos en la precisión y la recuperación ya que consideramos que en este problema de reconocer las caras son las más importante ya que nos parece crucial que al menos detecte todas las caras (recall) y que sea medianamente creíble cuando afirma que es un positivo(precisión). Además graficaremos su curva ROC y pondremos a todos los modelos a detectar caras en una misma foto.

Árbol de decisión:

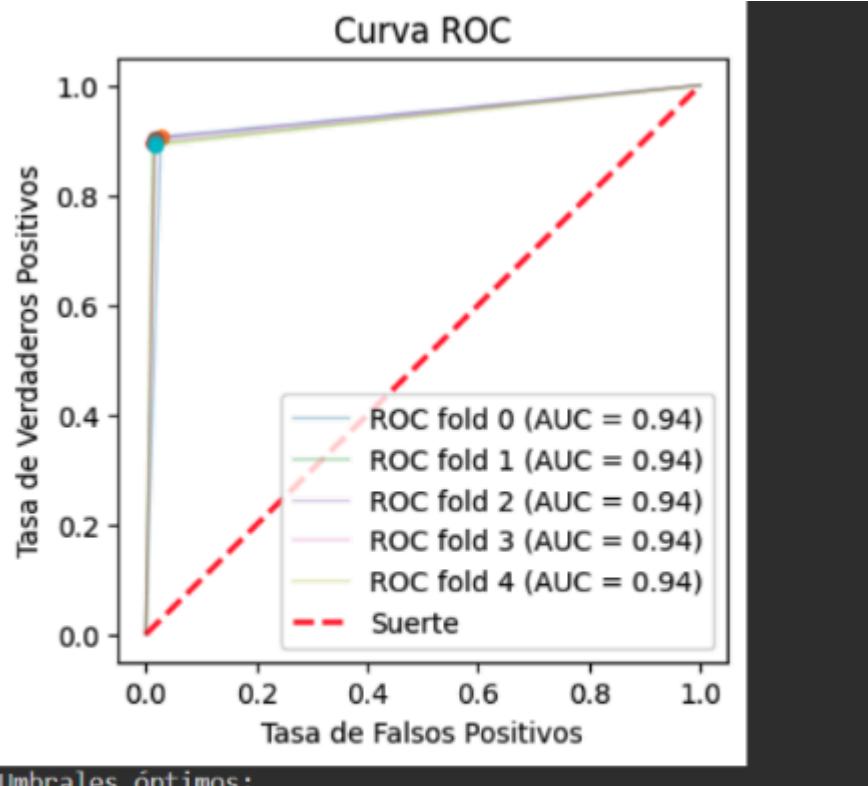
El problema es de clasificación binaria por lo cual en lo primero que pensamos es en un árbol de decisión ya que es el primer modelo que dimos para este tipo de problemas. Decidimos utilizar el modelo por default de sklearn ya que queríamos un árbol sin máxima profundidad y que usara Gini para medir la impureza ya que es el que usamos en clase. Dado que este es un modelo muy primitivo decidimos hacer uno solo de estos.

Los resultados fueron los siguientes:

Después de promediar el cv tuvimos estas métricas:

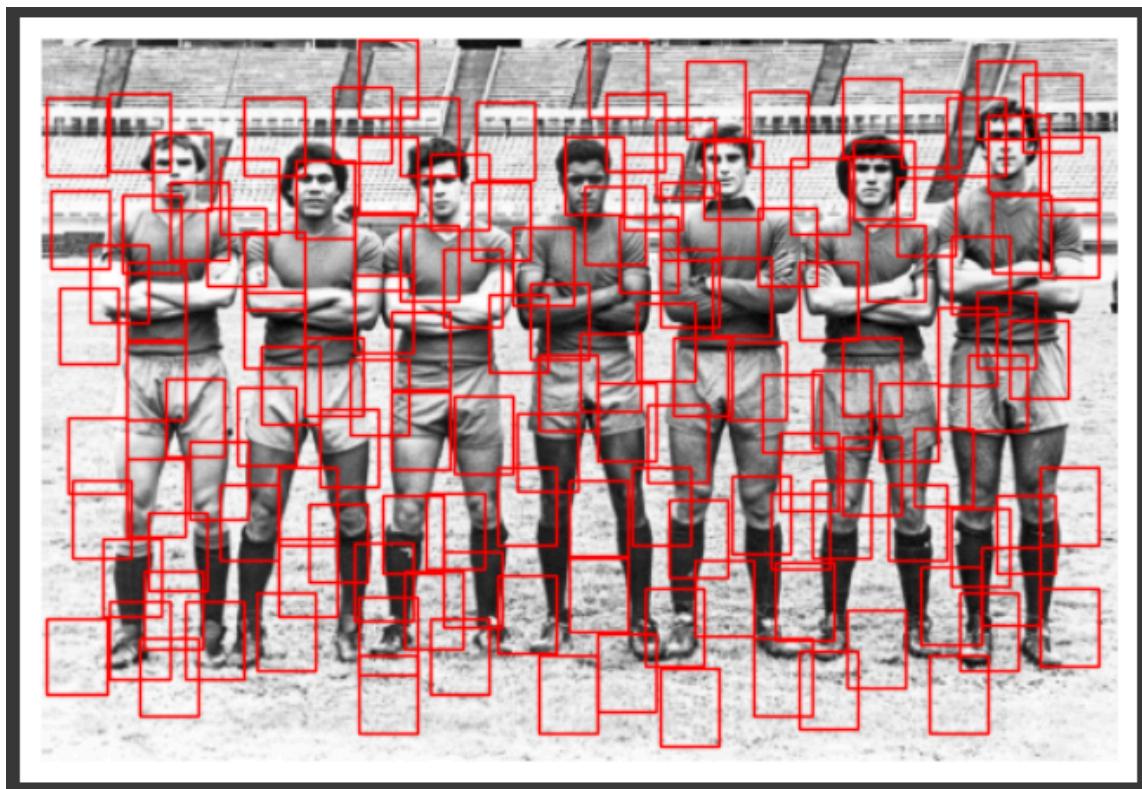
```
→ TIME          20.845238
ACCURACY       0.974522
PRECISION      0.917622
RECALL         0.940596
F1              0.928486
B_ACCURACY     0.940596
dtype: float64
```

Podemos ver que el recall es algo bajo lo que quiere decir que no logra detectar todas las caras lo cual es malo ya que lo ideal sería por lo menos estar arriba del 95% en el recall. Pero lo verdaderamente grave de este modelo es que la precisión es realmente baja rondando el 90 % que esta métrica nos indica qué tan creíbles somos cuando decimos que es positivo. Dadas estas métricas podemos graficar la curva ROC que queda la siguiente.



Umbral óptimos:
`[1.0, 1.0, 1.0, 1.0, 1.0]`
 Mejor umbral promedio: 1.0
 Desviación estándar del umbral: 0.0
 FPR promedio: 0.01746, TPR promedio: 0.8986599999999999
 AUC promedio: 0.94059580805045

Luego como último test visual pusimos al modelo a detectar caras en la siguiente foto:



Podemos apreciar que es bastante malo el modelo en una situación más realista en las cuales no son solo parches de 62*47. Esto nos sugiere inclinarnos por otro modelo.

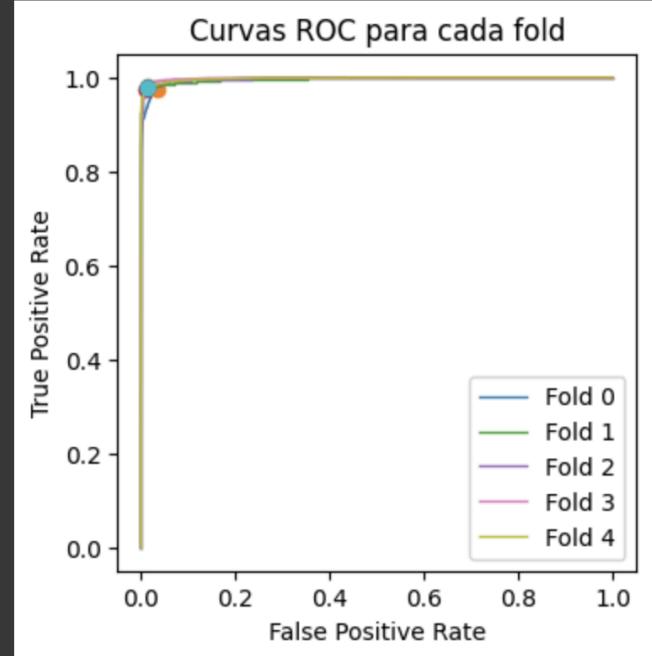
Regresión Logística:

El segundo modelo que utilizamos es el dado en clase el cual es una regresión logística la cual tiene regularización de Ridge. Las métricas son las siguientes:

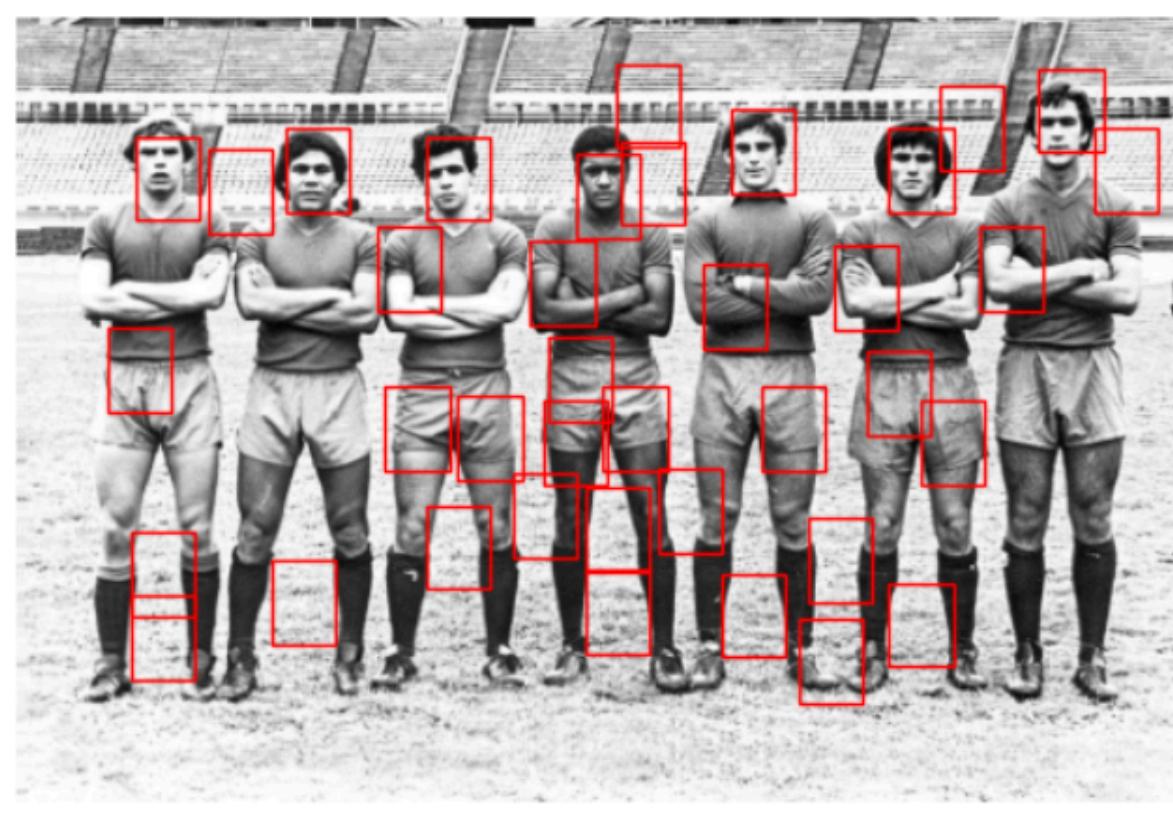
```
TIME           3.491952
ACCURACY       0.989927
PRECISION      0.974641
RECALL          0.968713
F1              0.971162
B_ACCURACY     0.968713
AUC             0.997136
TPR             0.942492
FPR             0.005066
dtype: float64
```

Este modelo mejora en todos los sentidos al anterior, hasta el tiempo de entrenamiento es mejor. Sin contar que la precisión y el recall están por encima del 95%. Estás satisfacen en gran medida lo que buscamos además de ver una mejora significativa en la curva ROC que mostraremos a continuación:

```
Umbrales óptimos:
[0.2615, 0.1671, 0.1309, 0.1049, 0.1177]
Mejor umbral promedio: 0.15642
Desvío umbral: 0.05649900530097854
FPR promedio: 0.01664, TPR promedio: 0.9773
```



Vemos en la curva que los umbrales se aproximan más a lo que buscamos que se acerca al 0.15. Podemos ver más claramente esta mejora a nivel visual con la siguiente imagen:



Podemos percatarnos de que en la imagen la accuracy de los rostros es casi perfecta por lo cual nos acercamos más a lo que queremos.

Boosting:

Ahora empezaremos con los ensambles en nuestra búsqueda de algún modelo un poco más preciso a nivel empírico ya que los modelos simples no satisfagan nuestras expectativas.

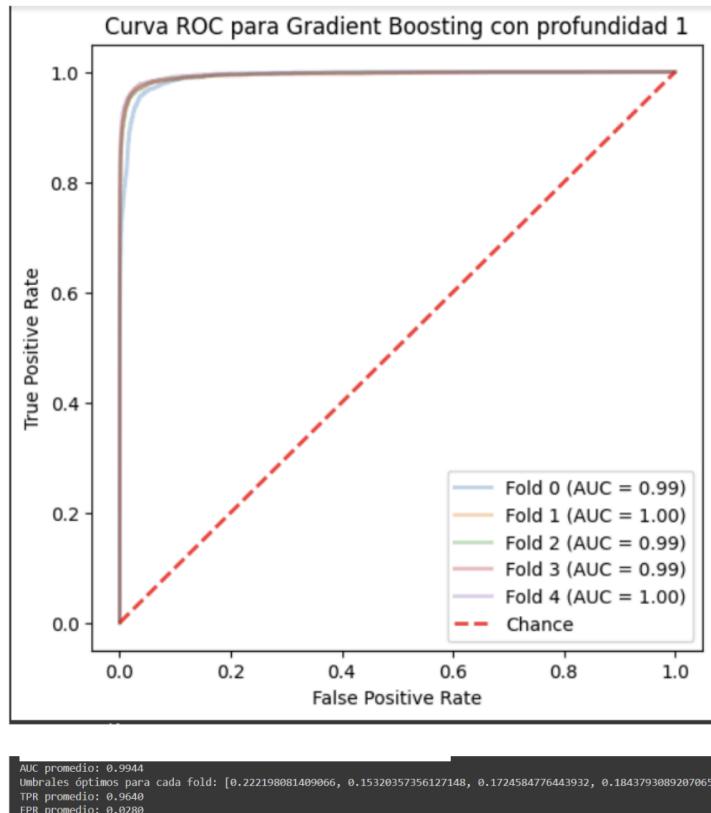
Gradient Boosting:

Para gradient boosting utilizamos un solo modelo que tiene árboles de profundidad

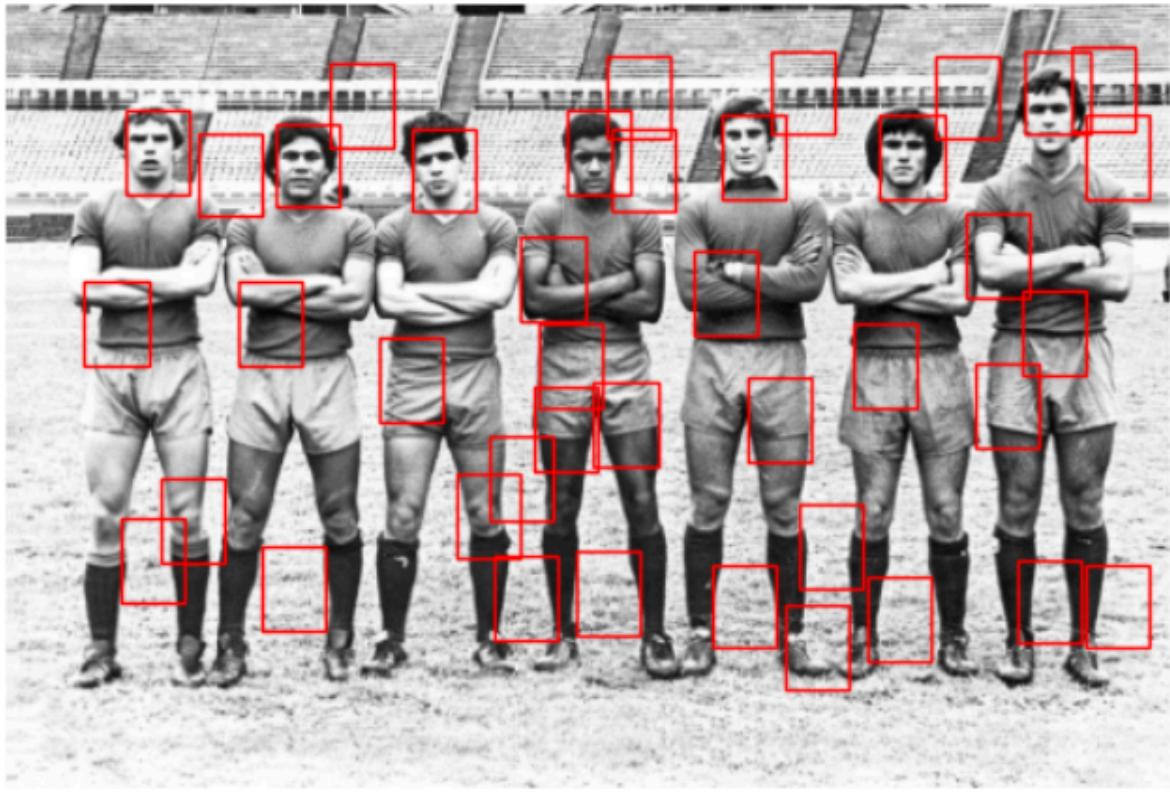
1. Los resultados son los siguientes:

```
fit_time      191.327455
score_time     0.121349
test_acc       0.979695
test_prec      0.972050
test_rec       0.909153
test_f1        0.937403
test_b_acc     0.909153
dtype: float64
```

Podemos observar que no solo empeora el tiempo de entrenamiento de manera considerable (era esperable ya que es ensemble) si no que también empeora las métricas en general. Teniendo un recall incluso peor que el árbol de decisión.



A nivel de la curva ROC no se pueden observar cambios demasiado significativos pero teniendo un área debajo de la curva similar. Pero la prueba de la foto sigue siendo determinante para determinar que necesitamos seguir buscando.



Esto nos puede dar indicios de que los ensembles pueden mejorar la regresión logística pero quizás gradient boosting no es el camino a seguir. Por eso decidimos probar varios modelos de ADA Boosting.

ADA Boosting:

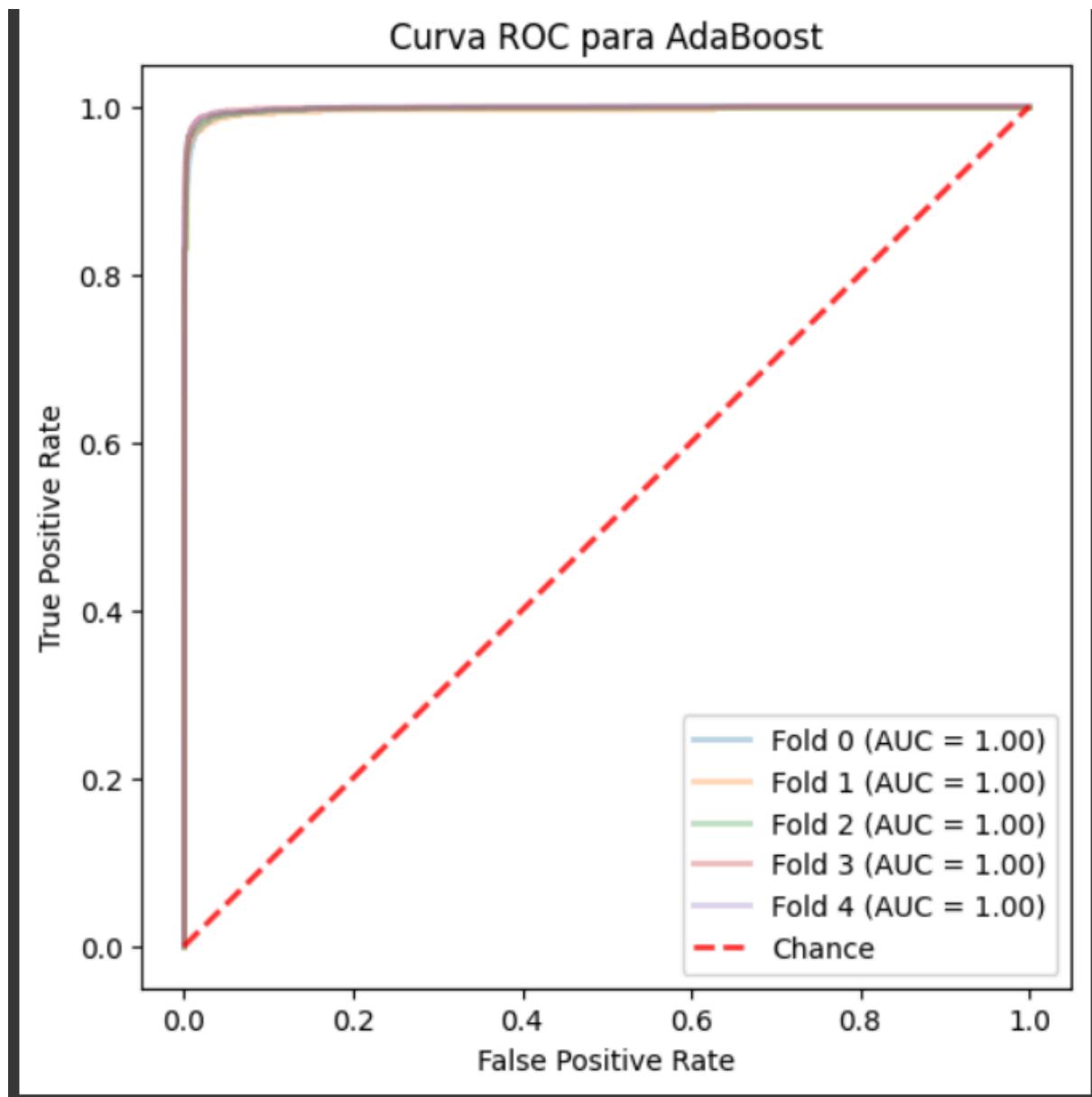
Para ADA Boosting decidimos hacer 3 modelos: 2 de ellos con estimador base un árbol de decisión y otro con regresión logística como base.

Modelo 1:

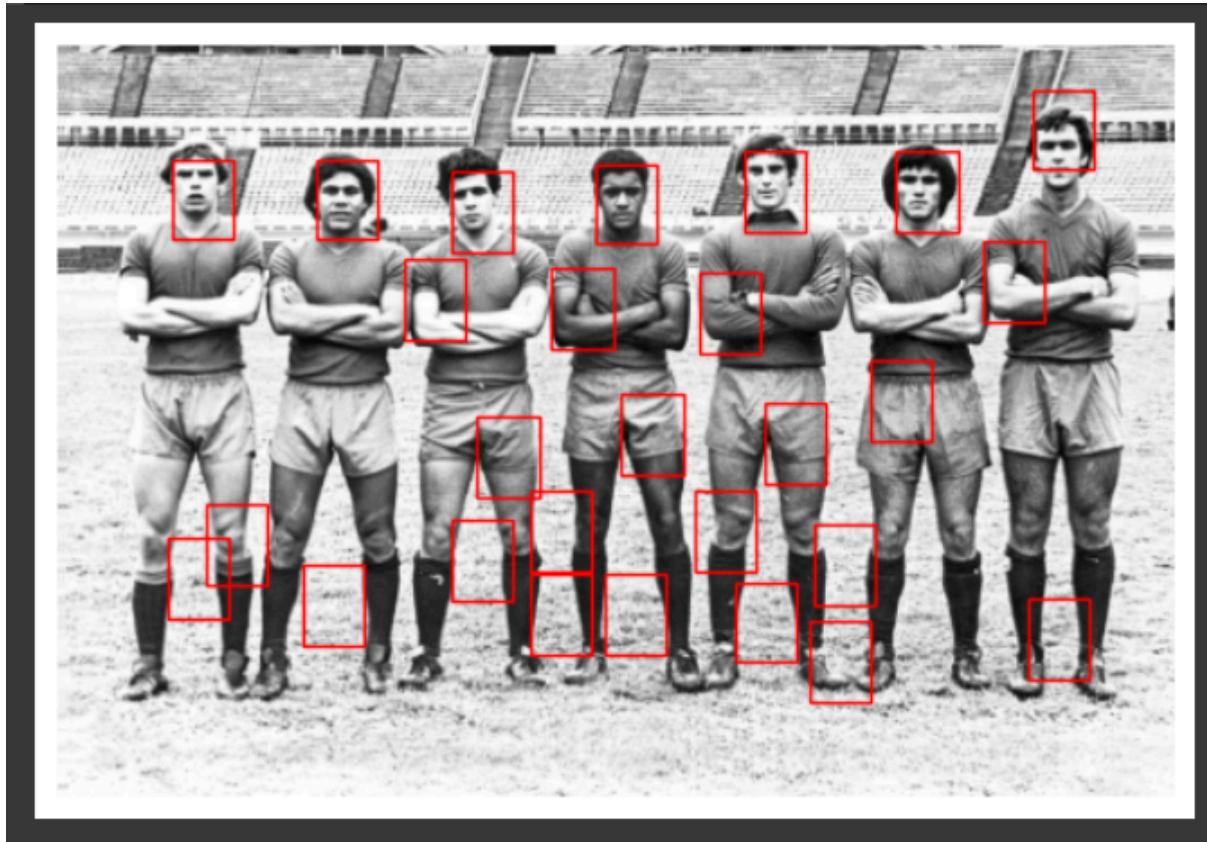
Nuestro primer modelo de ADaBoosting lo hicimos con 200 estimadores y un árbol de decisión de máxima profundidad 1. Las métricas son las siguientes:

```
fit_time      435.570201
score_time    1.516347
test_acc      0.990533
test_prec     0.969366
test_rec      0.976855
test_f1       0.972940
test_b_acc    0.976855
dtype: float64
```

Las métricas siguen siendo muy parecidas como la curva ROC.



Además empíricamente no queda tan mal:

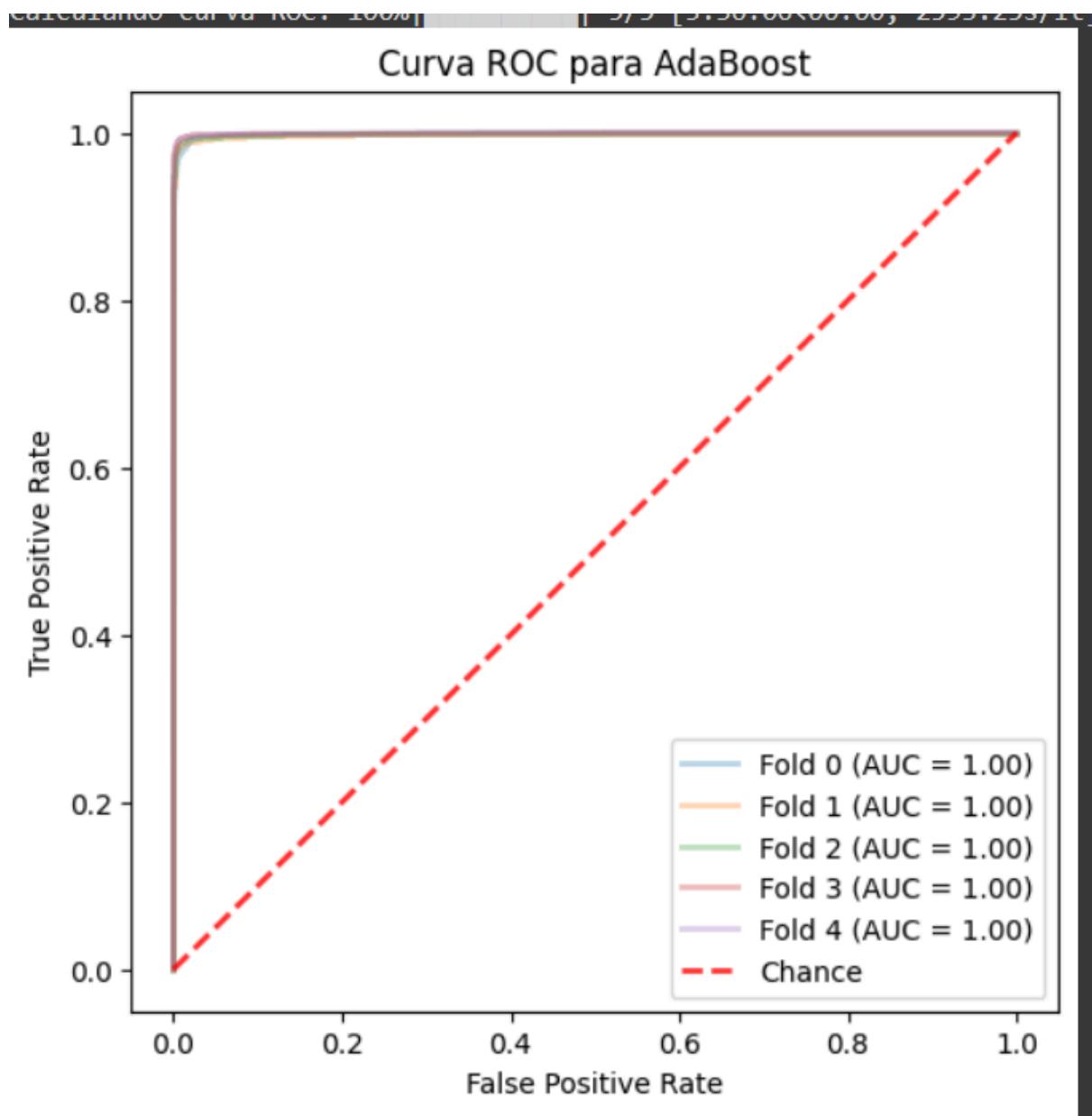


Modelo 2:

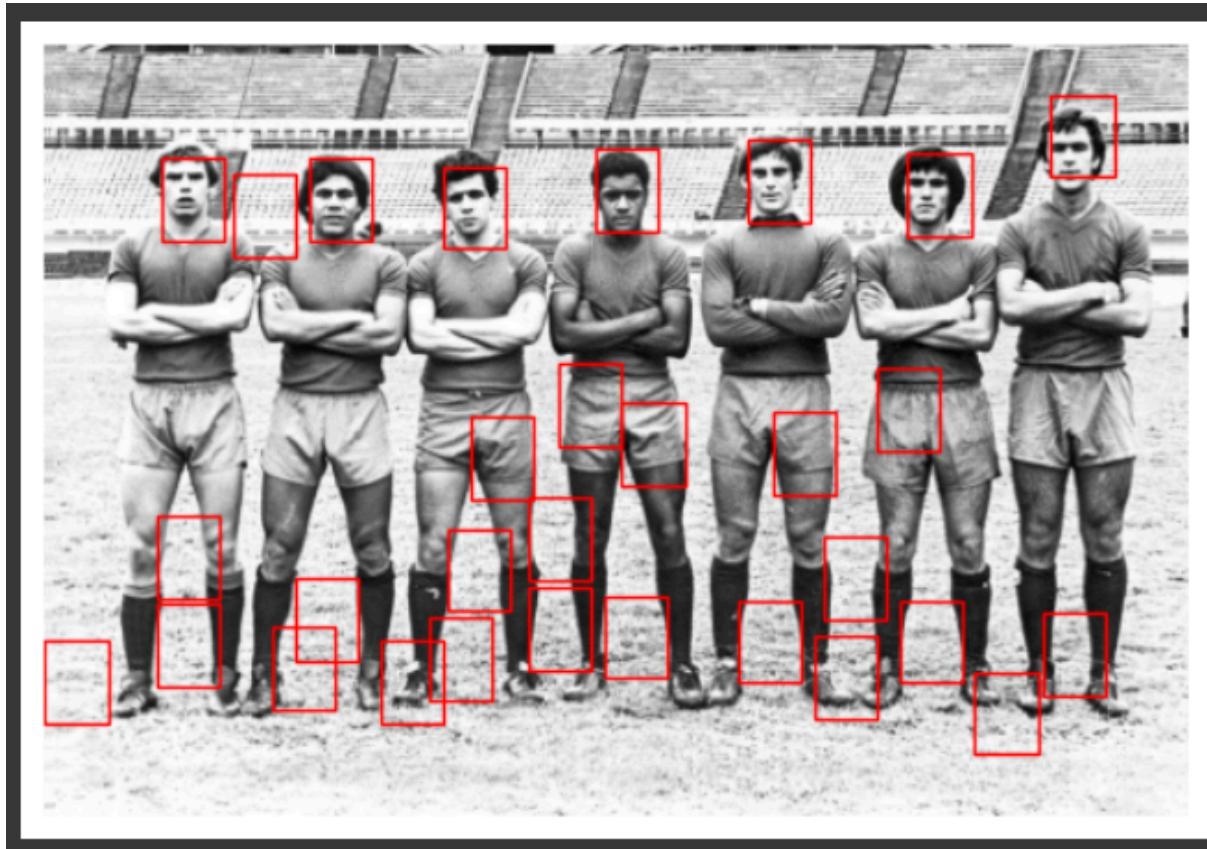
Probamos el mismo modelo anterior pero con árboles de profundidad máxima 6 en lugar de 1. Las métricas quedaron de la siguiente manera.

```
fit_time      2379.136431
score_time     1.855655
test_acc       0.995014
test_prec      0.992038
test_rec       0.979028
test_f1        0.985394
test_b_acc     0.979028
dtype: float64
```

La curva ROC queda igual a las otras:



Con la foto sigue teniendo defectos pero mejora bastante:

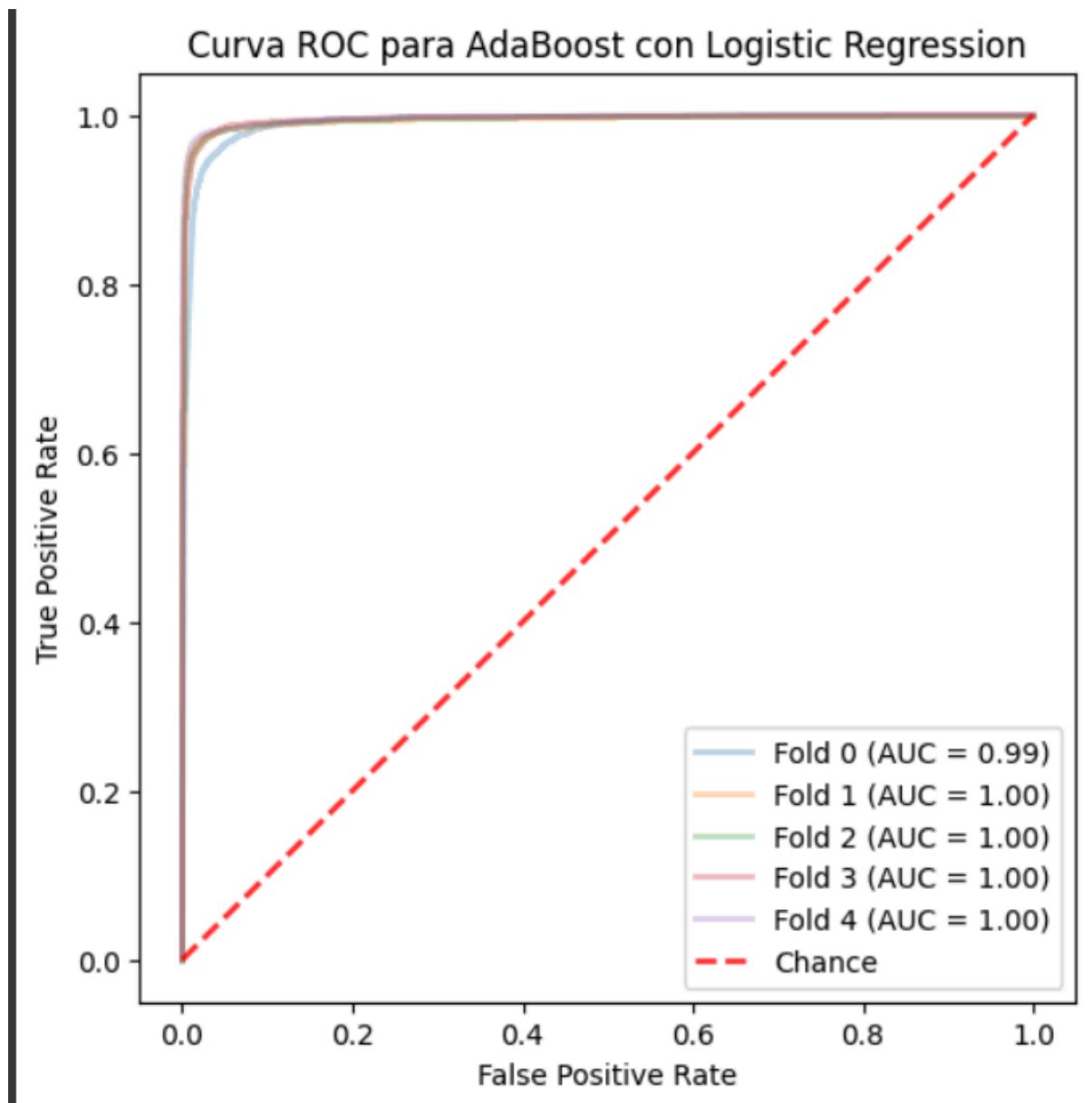


Modelo 3:

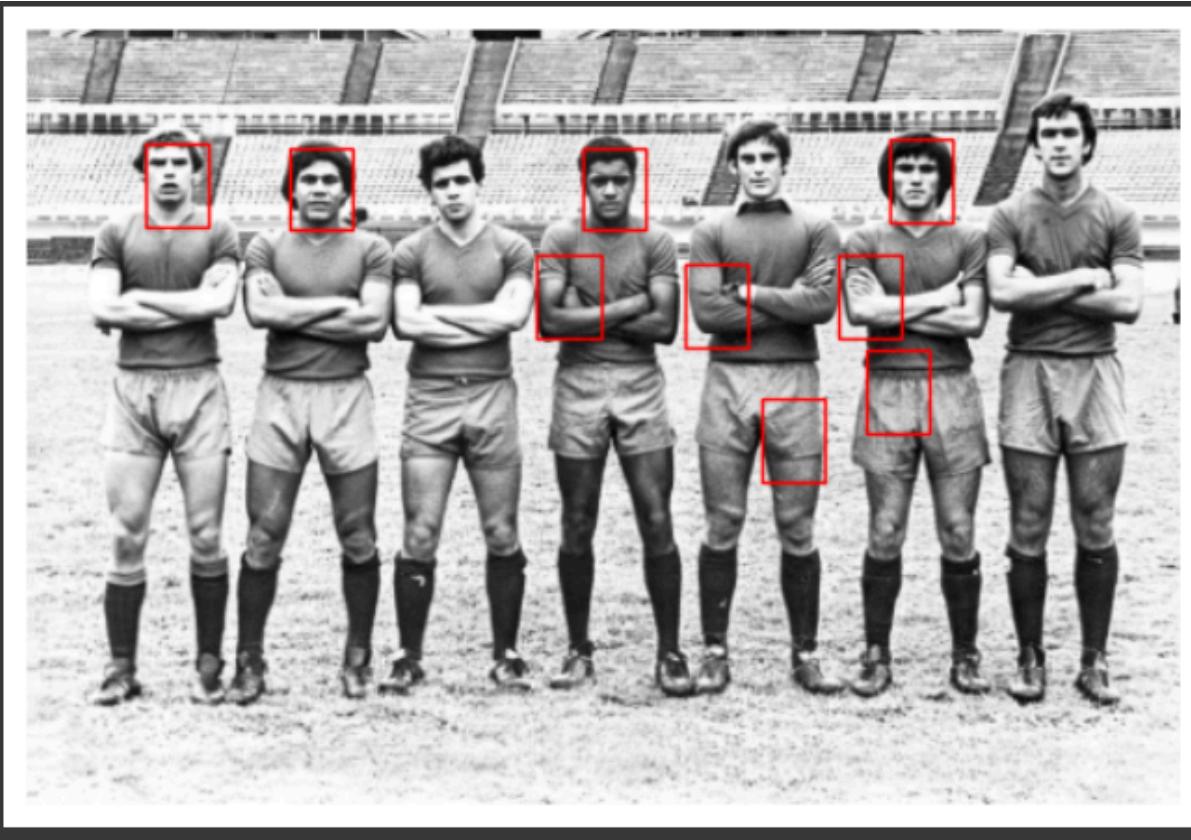
ADA Boosting con logistic regression de 100 estimadores este tuvo un resultado en metricas que dieron bastante mal con el peor recall posible;

```
fit_time      39.037485
score_time    1.207707
test_accuracy 0.971556
test_precision 0.977659
test_recall   0.719639
test_f1       0.828582
dtype: float64
```

Sin embargo la curva ROC nos da muy bien:



Pero a nivel empírico con la foto de prueba es el que menos tiene fallos es decir que predice la menor cantidad de falsos positivos pero también le falta reconocer algunas caras. Este sería un buen candidato si no hubiera fallado tanto el recall.



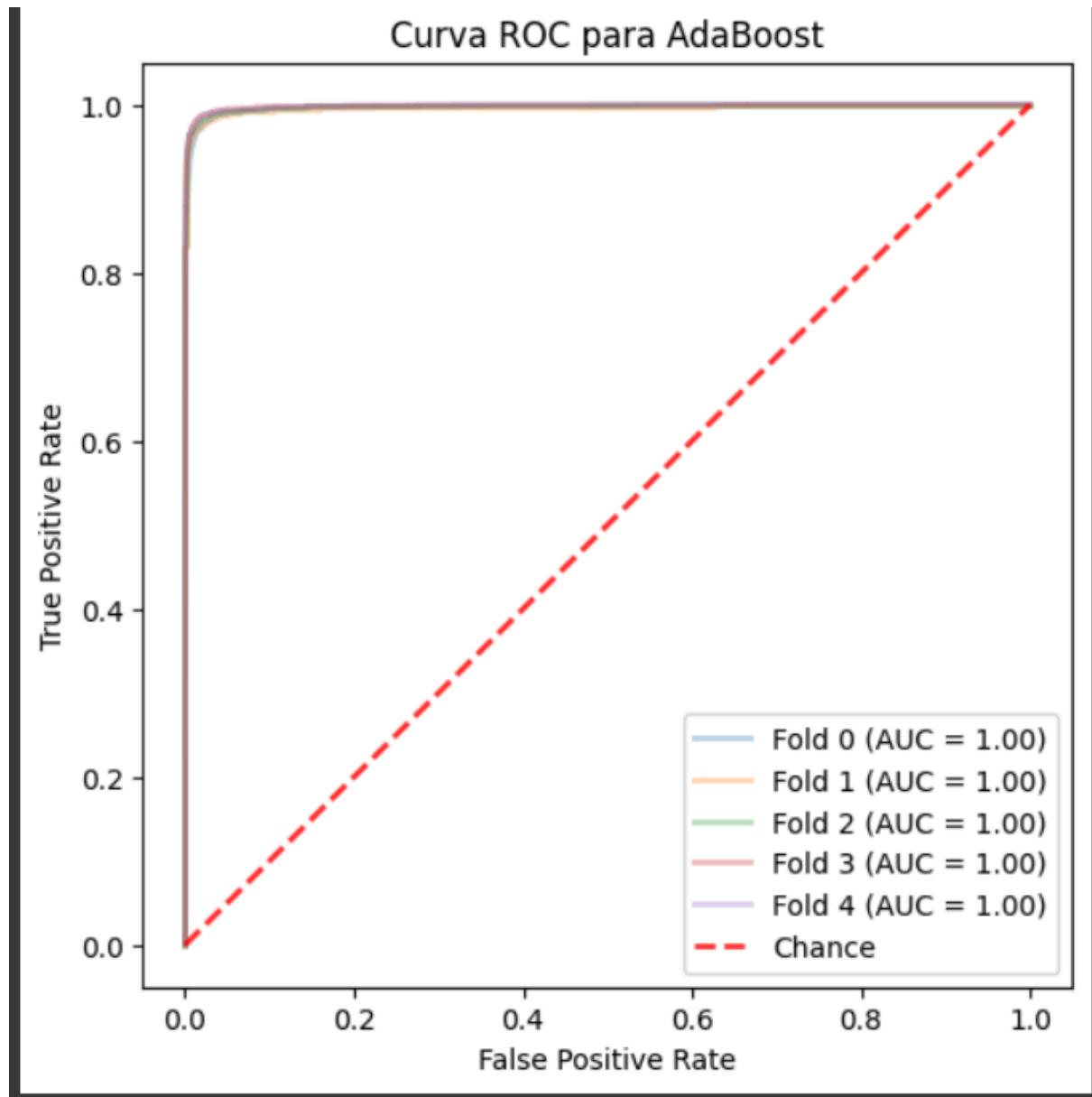
Random Forest:

Modelo 1:

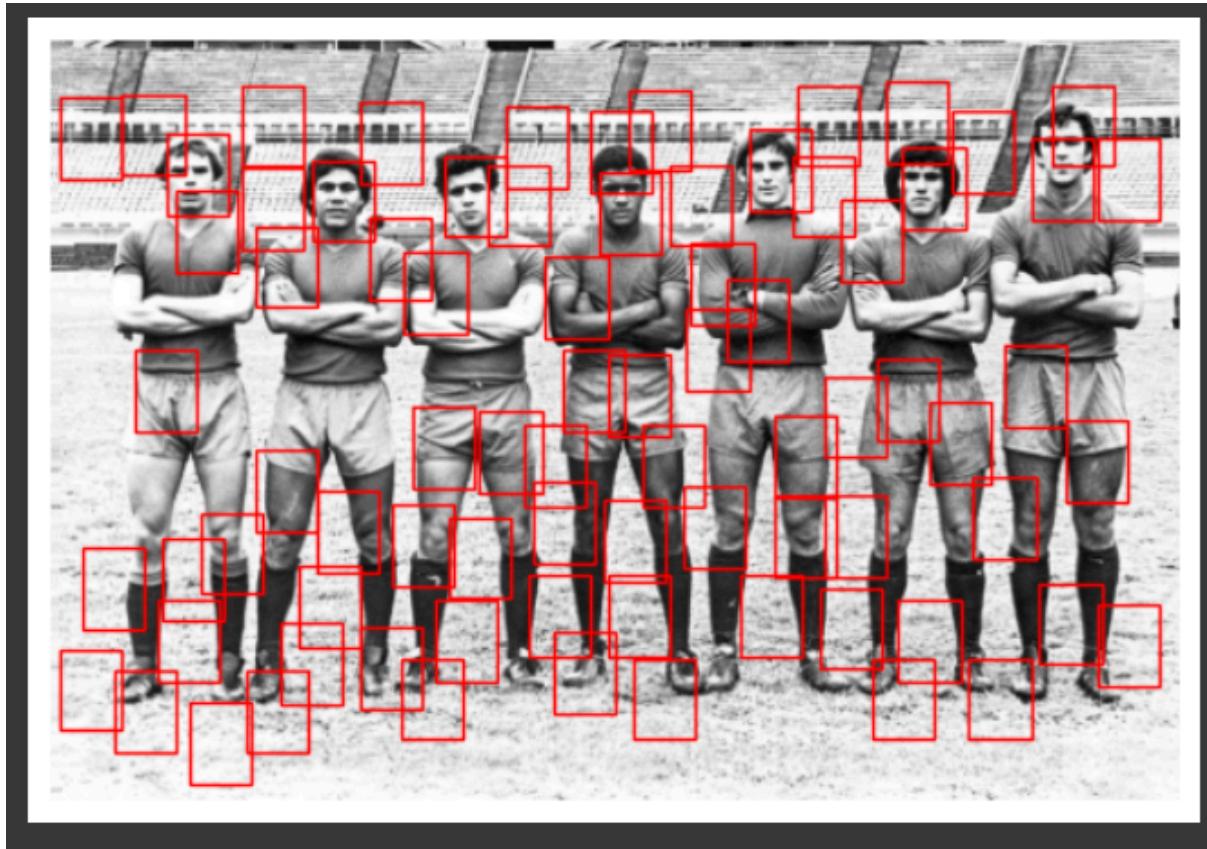
Para terminar con los ensambles hicimos 2 modelos de random forest. El primer con 10 estimadores con árboles de maxdepth tiene los siguientes resultados:

```
fit_time      13.659474
score_time    0.111507
test_acc      0.988246
test_prec     0.984034
test_rec      0.947236
test_f1       0.964673
test_b_acc    0.947236
dtype: float64
```

Vemos que los resultados de las métricas son muy buenos teniendo como destacado una precisión bastante buena. La curva ROC se ve de esta manera.



Pero empíricamente sigue teniendo problema:

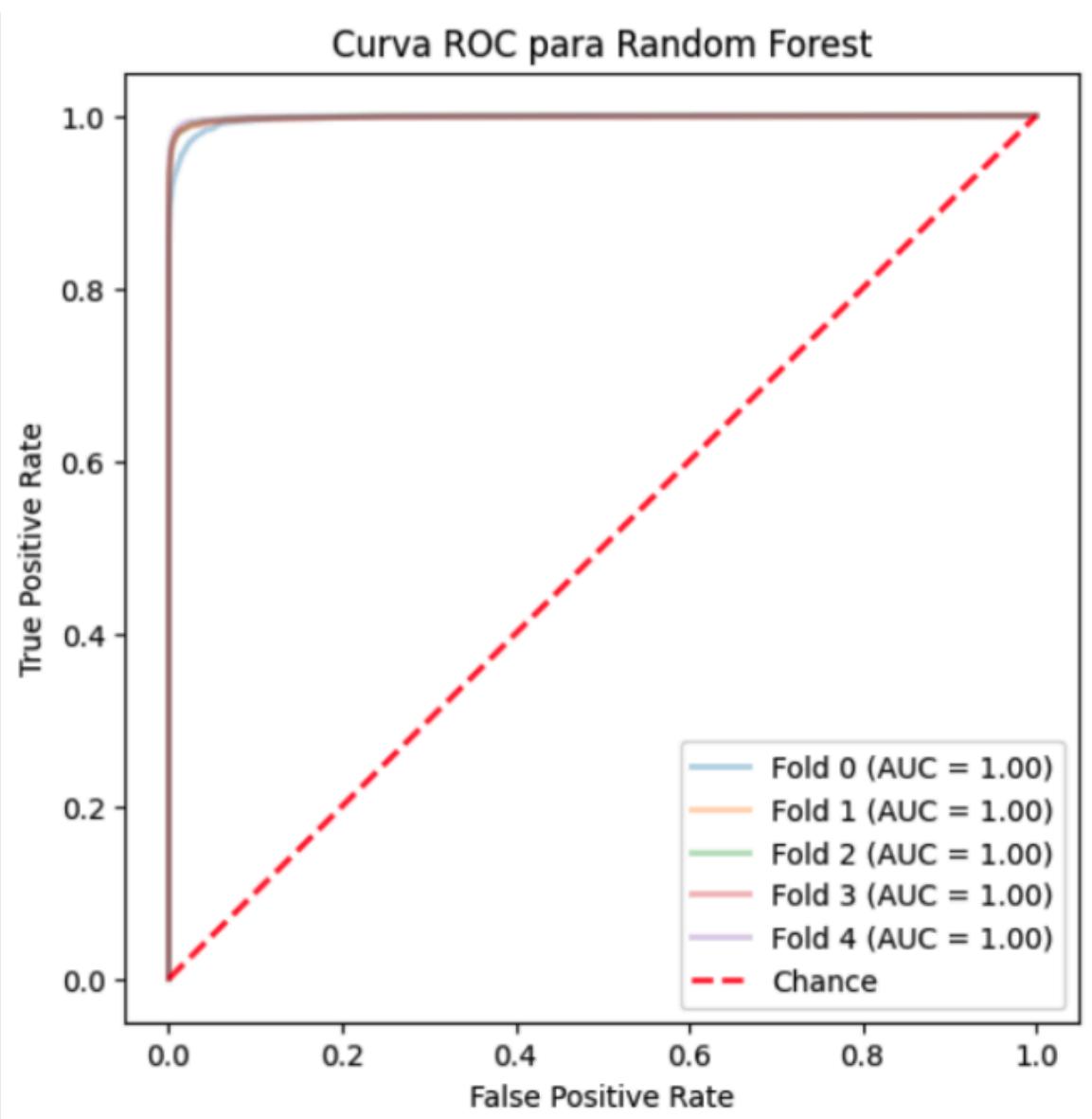


Modelo 2:

Como segundo modelo de random forest probamos un modelo con muchos más estimadores(10 veces más) esperando mejores resultados.

```
fit_time      130.824295
score_time     0.495673
test_acc       0.991060
test_prec      0.987573
test_rec       0.960417
test_f1        0.973440
test_b_acc     0.960417
dtype: float64
```

Vemos que a nivel de métricas se ve una pequeña mejora sobre todo en el f1-Score que es un compromiso entre la precisión y el recall. Pero la curva ROC tiene una leve mejoría casi imperceptible.



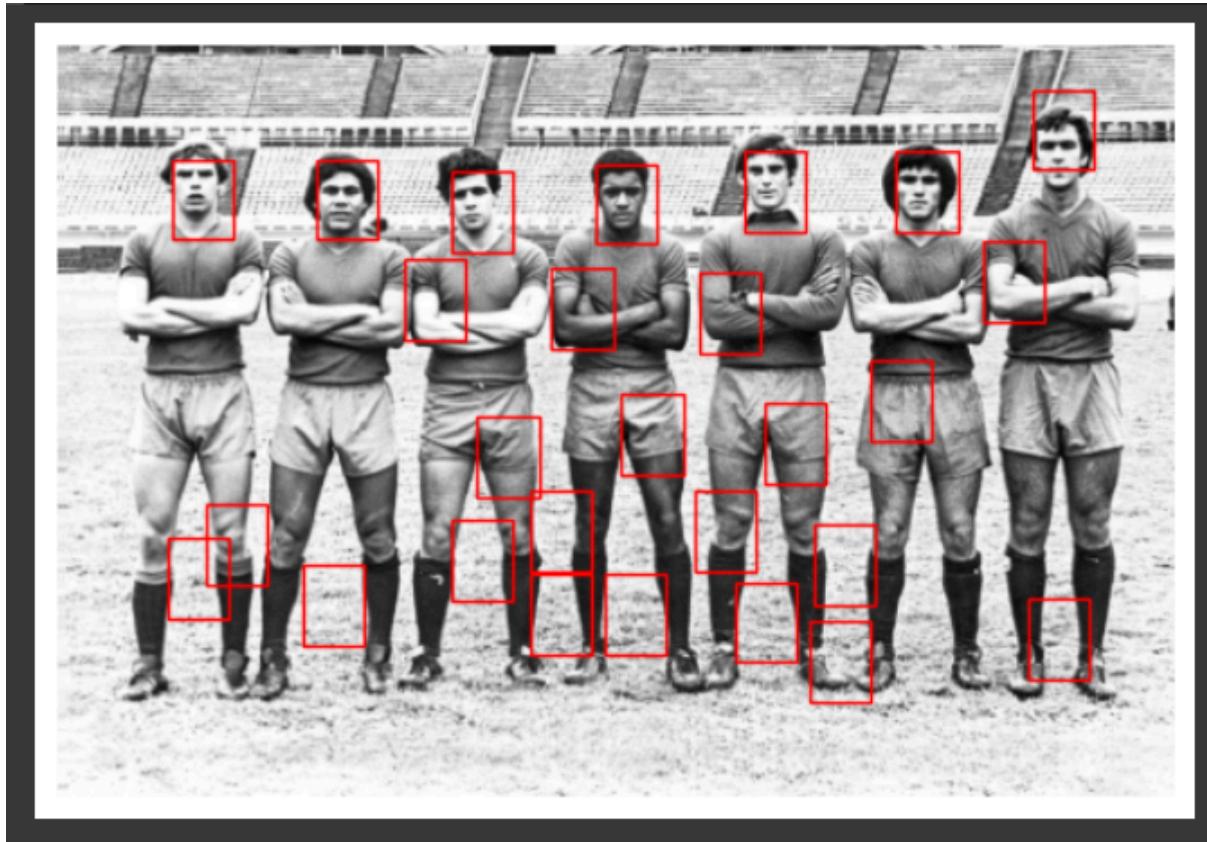
AUC promedio: 0.9982

Umbrales óptimos para cada fold: [0.31, 0.24, 0.23, 0.2, 0.2]

TPR promedio: 0.9801

FPR promedio: 0.0140

Sin embargo cuando hacemos la prueba con la foto separada vemos una mejoría significativa como para elegirlo por encima del random forest de 10 estimadores ya que tiene muchos menos fallos por más que su tiempo de entrenamiento fue mucho mayor.



Vemos que en este caso también detectó todas las caras si bien hubo muchos fallos igual.

Redes neuronales:

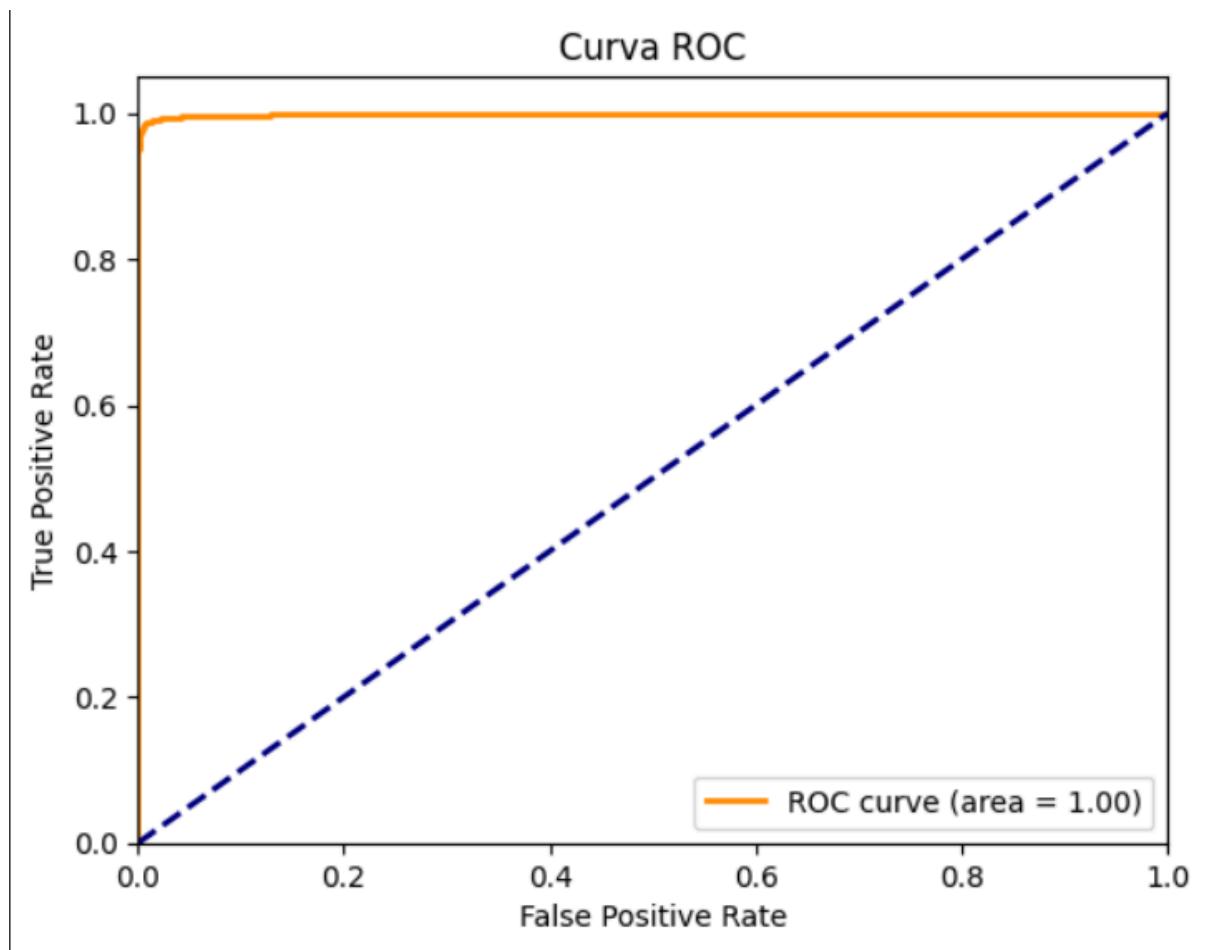
Para modelos de redes neuronales probamos 2 modelos:

Modelo 1:

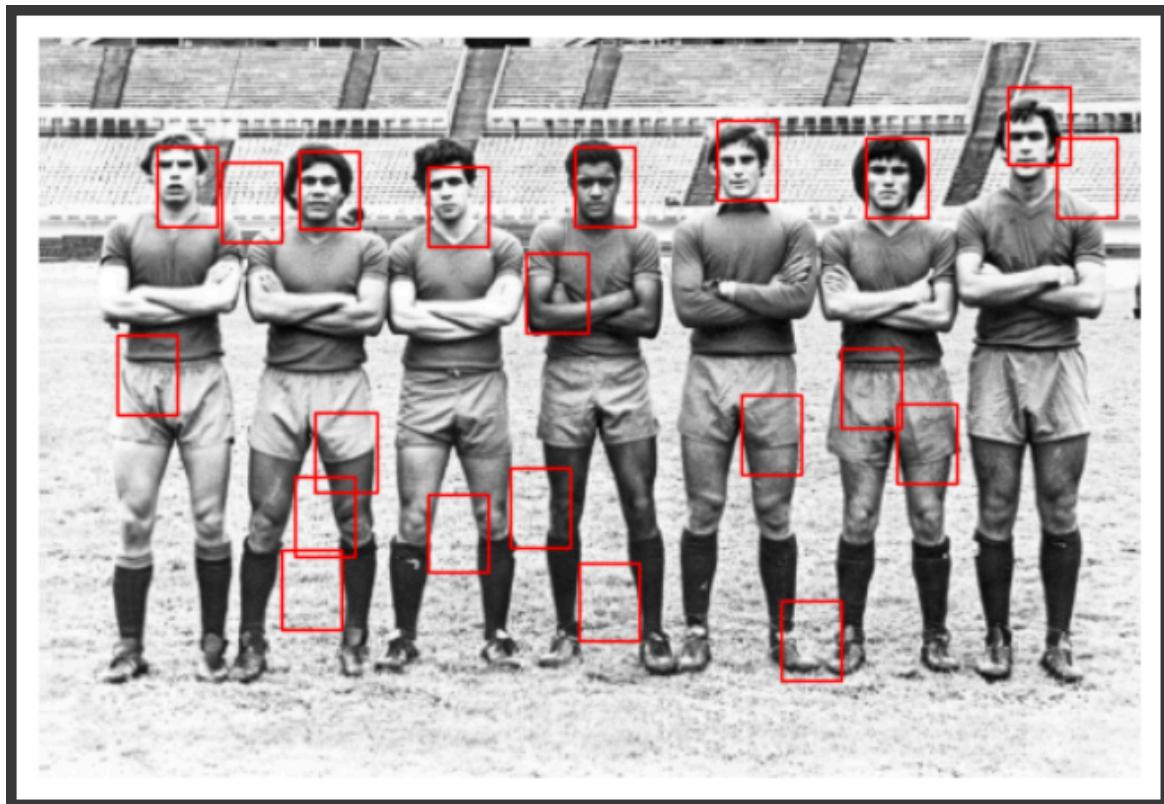
Un primer modelo con 2 capas internas de 64 y 32 neuronas, un tamaño de batch de 32 y 10 épocas. Que nos dio las métricas siguientes:

	precision	recall	f1-score
0.0	1.00	1.00	1.00
1.0	0.96	0.98	0.97
accuracy			0.99
macro avg	0.98	0.99	0.98
weighted avg	0.99	0.99	0.99

Una curva ROC prácticamente perfecta.



Sin embargo la foto sigue teniendo problemas.

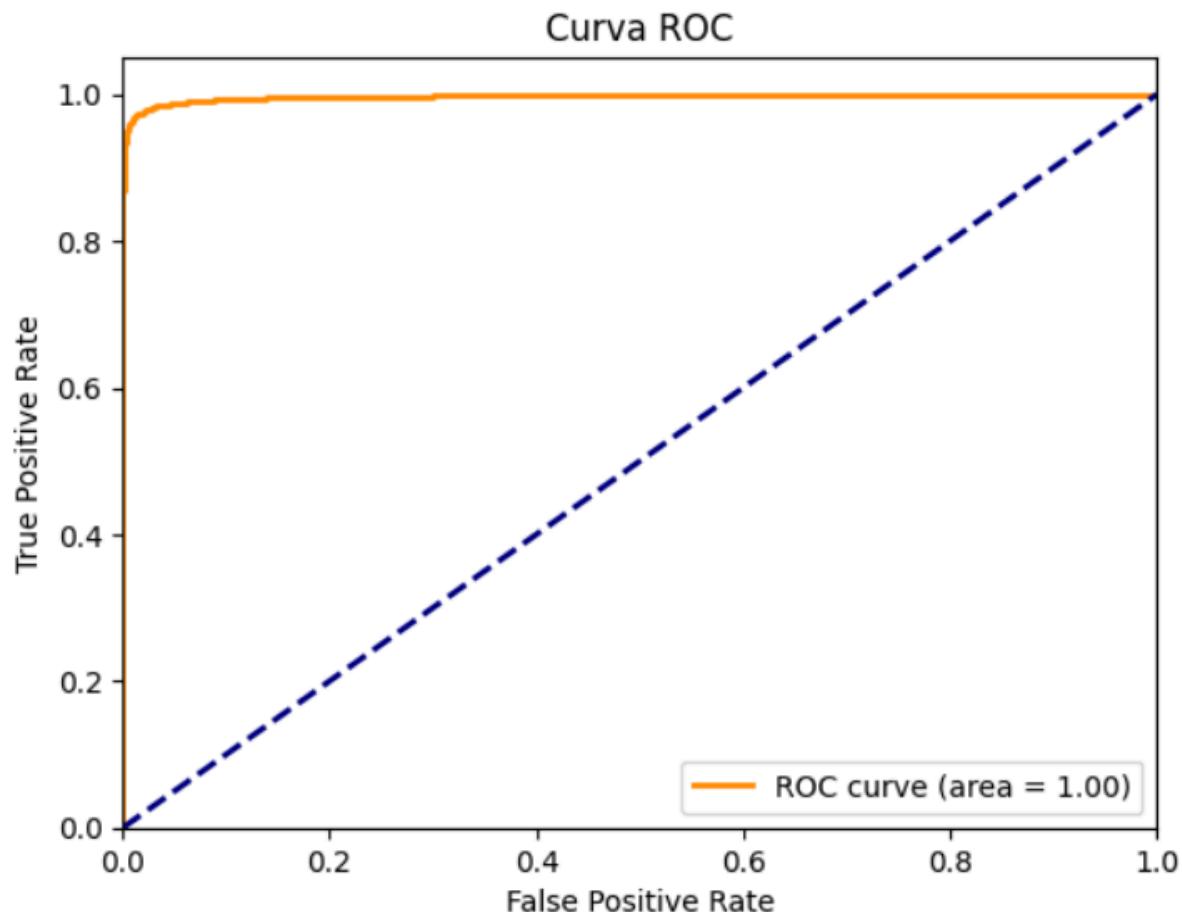


Modelo 2:

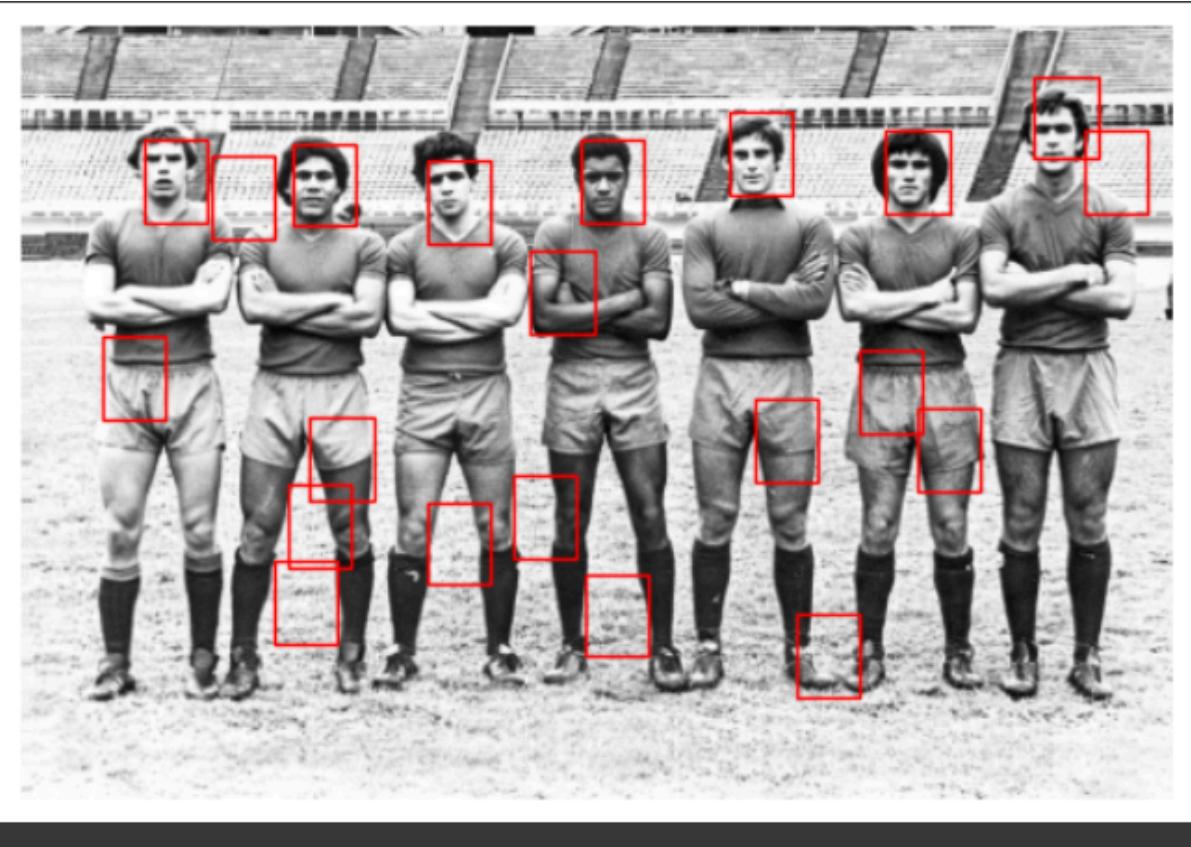
Para nuestro segundo modelo de redes neuronales decidimos hacer una red mucho más grande de 3 capas internas una de 128 con una capa de dropout de 0.5 después una de 64 y otra 32. Que tiene las siguientes métricas

Evaluación en el conjunto de prueba para el segundo modelo:				
	precision	recall	f1-score	support
0.0	0.99	1.00	1.00	25080
1.0	0.97	0.94	0.95	2638
accuracy			0.99	27718
macro avg	0.98	0.97	0.97	27718
weighted avg	0.99	0.99	0.99	27718

La curva ROC es la siguiente que también es muy buena.



Pero la foto sigue teniendo los mismos problemas, es más predice lo mismo que la red neuronal anterior con menor cantidad de épocas y menos cantidad de neuronas.



Para elegir el modelo nos basaremos como ya dijimos anteriormente en la precision y el recall, tambien tomaremos muy en cuenta las pruebas empiricas ya que las curvas ROC y las metricas dieron muy parecidas. Despues de una discusión nos quedaron 3 candidatos claros. El primero seria el ADA Boost con logistic regression que por mas que sus metricas fueron bastante malas fue el que empiricamente reconocio menos fondos como caras. Este lo descartamos porque si bien fue el que menos fallo tambien no reconocio el 3 caras de las 6. El siguiente candidato claro fue el segundo random forest que tenia 100 estimadores el cual si tiene buenas metricas y reconoce todas las caras de manera empírica pero tambien reconoce muchos fondos. El ultimo modelo y por el cual nos decantamos fue la primera red neuronal que tenia unas metricas muy parecidas a la segunda y la prueba empírica nos dieron exactamente igual, asi que nos decidimos por la primera por el menor tiempo de compilación.

En conclusión nuestro modelo elegido es nuestra primera red neuronal.

Elección del modelo:

Para elegir el modelo nos basaremos como ya dijimos anteriormente en la precisión y el recall, también tomaremos muy en cuenta las pruebas empíricas ya que las curvas ROC y las métricas dieron muy parecidas. Después de una discusión nos quedaron 3 candidatos claros. El primero sería el ADABOOST con logistic regression que por más que sus métricas fueron bastante malas fue el que empíricamente reconoció menos fondos como caras. Este lo descartamos porque si bien fue el que menos falló también no reconoció el 3 caras de las 6. El siguiente candidato claro fue el segundo random forest que tenía 100 estimadores el cual si tiene buenas métricas y reconoce todas las caras de manera empírica pero también reconoce muchos fondos. El último modelo y por el cual nos decantamos fue la primera red neuronal que tenía unas métricas muy parecidas a la segunda y la prueba empírica nos dieron exactamente igual, así que nos decidimos por la primera por el menor tiempo de compilación.

decantamos fue la primera red neuronal que tenía unas métricas muy parecidas a la segunda y la prueba empírica nos dieron exactamente igual, así que nos decidimos por la primera por el menor tiempo de compilación.

En conclusión nuestro modelo elegido es nuestra primera red neuronal.