

# COGNITO

María L. PÉREZ SAURA

Martín PIERANGELI

Marcelo M. RAPONI

Javier C. RODRÍGUEZ

MACHINE LEARNING  
UNSAM 2021



**HEART  
FAILURE  
PREDICTION**

# DATASET

## 299 PACIENTES 12 FEATURES 1 TARGET

### BINARIAS

**anaemia:** Disminución de glóbulos rojos o hemoglobina

**sex:** Mujer u hombre

**smoking:** Si el paciente fuma

**diabetes:** Si el paciente tiene diabetes

**high\_blood\_pressure:** Si el paciente tiene hipertensión

### CONTINUAS

**age:** Edad del paciente (años)

**creatinine\_phosphokinase:** Nivel de enzimas CPK en sangre (mcg/L)

**ejection\_fraction:** Porcentaje de sangre eyectada del corazón por contracción (%)

**platelets:** Plaquetas en sangre (kiloplatelets/mL)

**serum\_creatinine:** Nivel de creatinina sérica en sangre (mg/dL)

**serum\_sodium:** Nivel de sodio sérica en sangre (mEq/L)

**time:** Periodo de seguimiento (días)

### TARGET

**DEATH\_EVENT:** Si el paciente falleció durante el periodo de seguimiento

Fuente: <https://www.kaggle.com/andrewmvd/heart-failure-clinical-data>

# ALGUNAS OBSERVACIONES

**creatinine\_phosphokinase.mean()** = 581.8 mcg/L (valor normal <120 )

*Valores altos en pacientes con AC o Pericarditis posterior a un AC*

**ejection\_fraction.mean()** = 38% (valor mínimo crítico = 41%)

*La disminución puede deberse a un AC o Alta Presión*

**serum\_creatinine.mean()** = 1.4 mg/dL (valor normal: mujeres < 1.1, hombres < 1.3)

*Valores altos indican problemas renales, muy relacionado con la presión y el corazón*

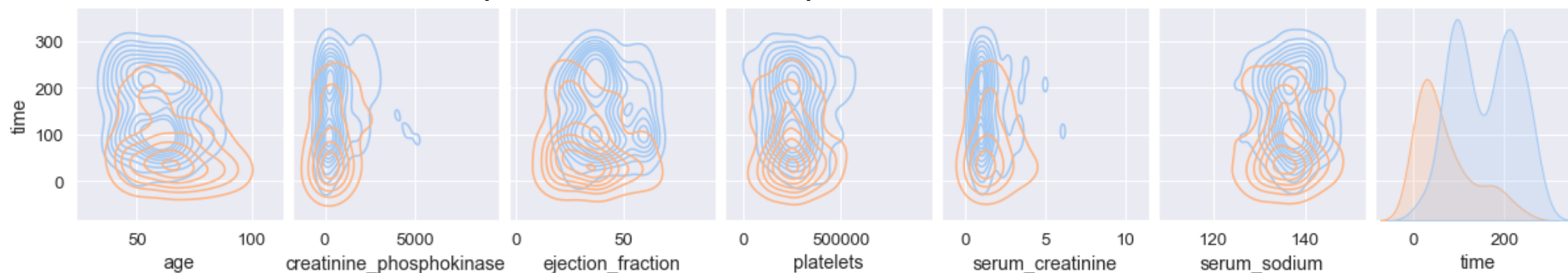
**serum\_sodium.mean()** = 136.7 mEq/L (valores normales 135-140)

**platelets.mean()** = 98k/mL (valor mínimo normal = 150)

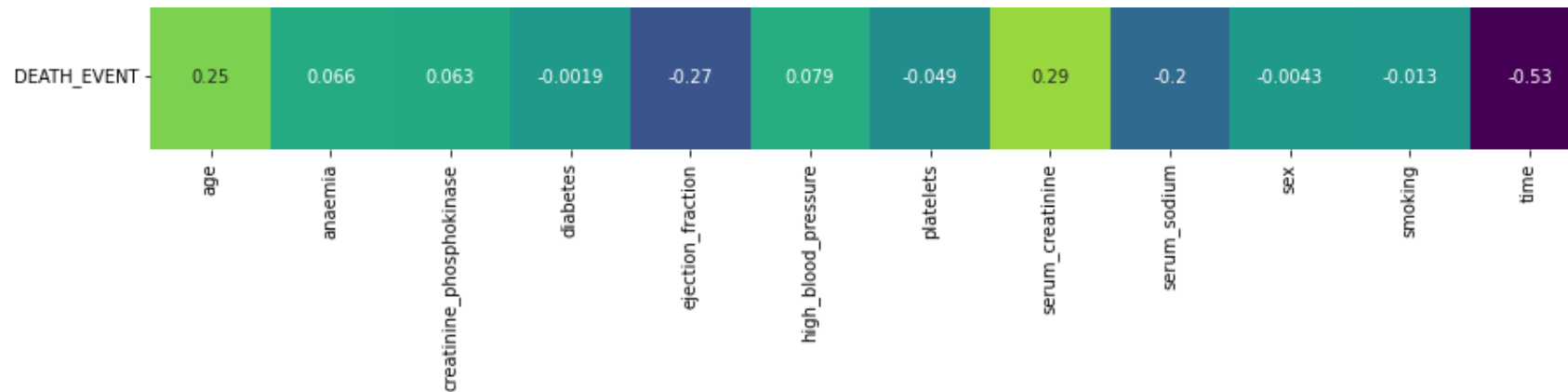
**En general, los valores están por fuera de lo normal, lo que indica que los pacientes sufren definitivamente complicaciones relacionadas.**

También se concluyó que el feature “time” era muy determinante y, en la práctica, indeterminable.

Por lo que se decidió dropear este feature.



## SELECCIÓN DE FEATURES



De la matriz de correlación reconocemos que los features binarios (patologías de base) son poco determinantes del target.

Para confirmar la hipótesis y descartarlos utilizamos RFECV:

RFECV o “Recursive Feature Elimination with Cross-Validation” es una herramienta de `sklearn.feature_selection` que verifica las `feature_importance` de un Random Forest grande en validación cruzada.

```
('age', True) ('anaemia', False)
('creatinine_phosphokinase', True)
('diabetes', False)
('ejection_fraction', True)
('high_blood_pressure', False)
('platelets', True) ('serum_creatinine',
True) ('serum_sodium', True) ('sex',
False) ('smoking', False) ('time', True)
```

# PREPROCESADO FINAL

Con los 6 features decididos (variables continuas sin time) y los datos de entrenamiento recortados:

Primero estandarizamos los datos usando **StandardScaler()** del módulo de *sklearn.preprocessing* ya que para la mayoría de los algoritmos de clasificación es casi una condición necesaria.

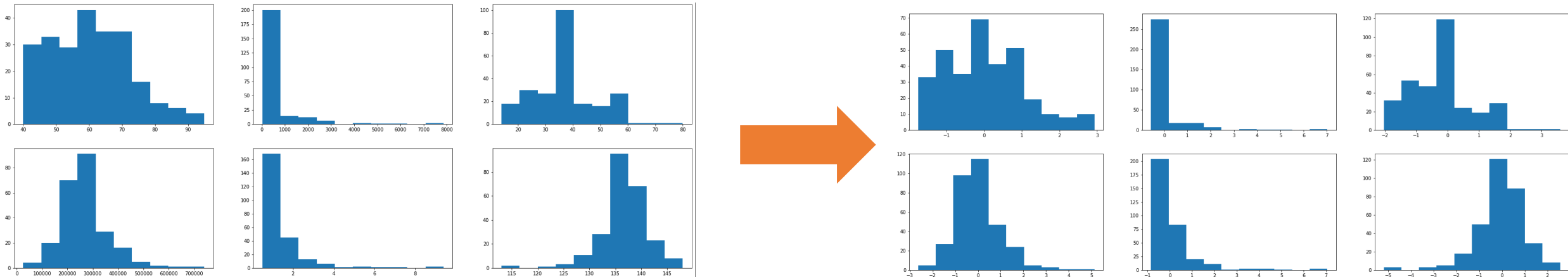
Luego decidimos realizar un **oversampling** de los datos por 2 motivos:

- Desbalance de clases target (163 “0s” vs 76 “1s”).
- Dataset chico sumado a los splits de testeo.

Entonces aplicamos la función **SMOTE**.

SMOTE es un algoritmo de oversampleo de la libreria de imblearn (imbalanced learn) desarrollada por el MIT

Como resultado, obtenemos un dataset balanceado 50-50 de 326 samples.



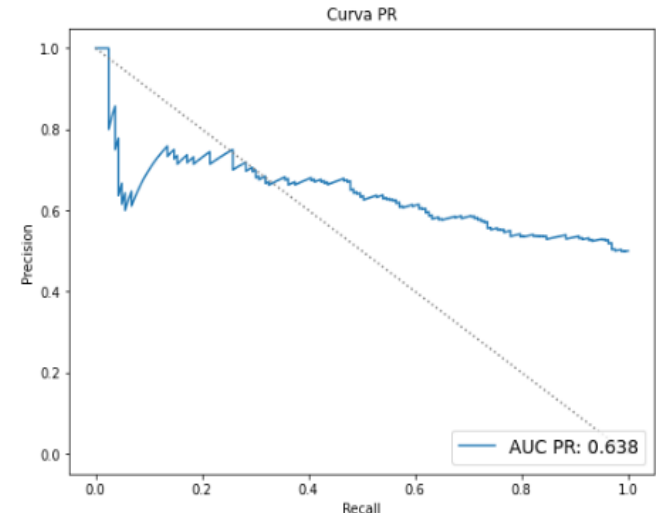
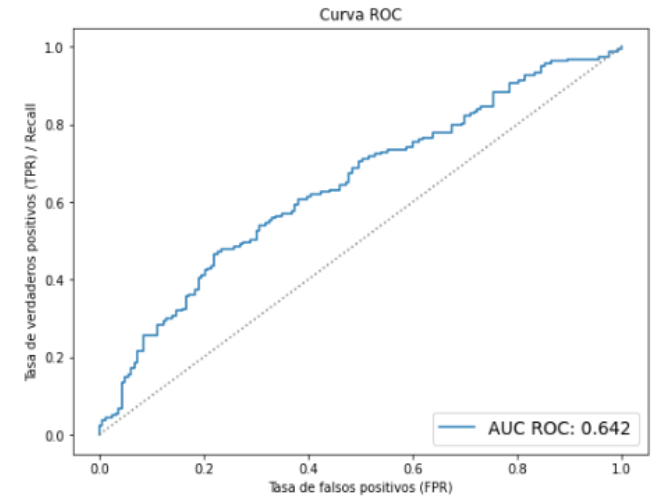
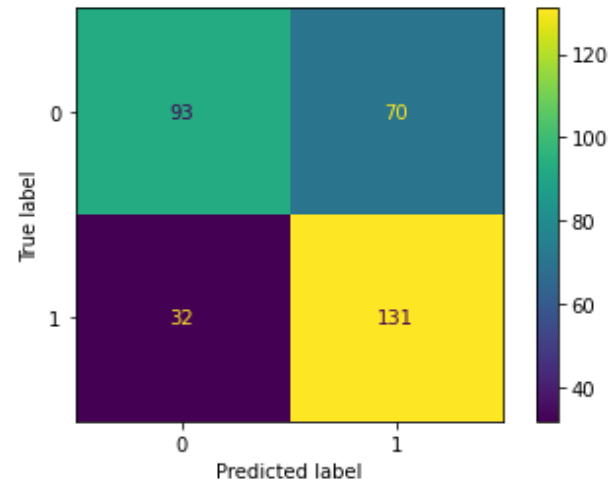
# BASELINE

Definimos la métrica a utilizar: **RECALL**

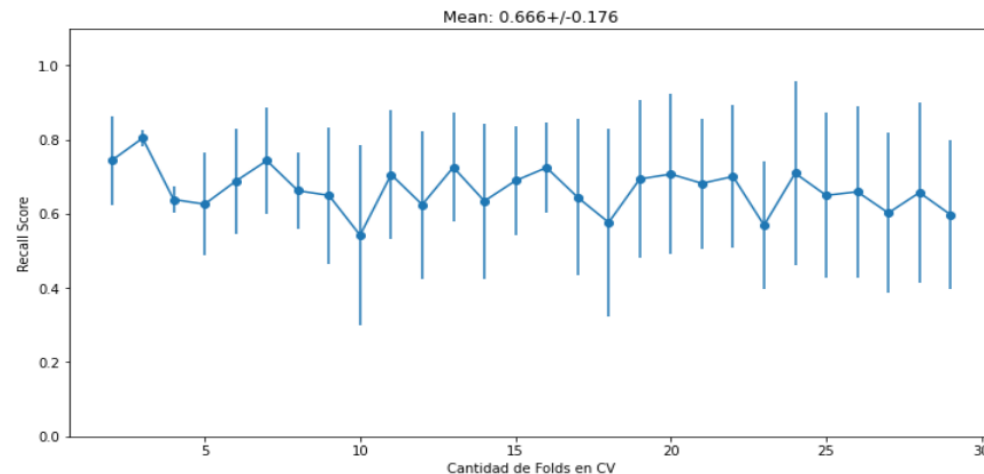
Queremos reducir los FN, es decir, pacientes que diríamos que NO tienen riesgo pero SI lo tienen en realidad.

## Perceptron

Cross Validation Score: 0.542 +/- 0.242				
	precision	recall	f1-score	support
0	0.74	0.57	0.65	163
1	0.65	0.80	0.72	163
accuracy			0.69	326
macro avg	0.70	0.69	0.68	326
weighted avg	0.70	0.69	0.68	326



Observamos que el score tiene muy alta desviación



# Perceptron Optimizado

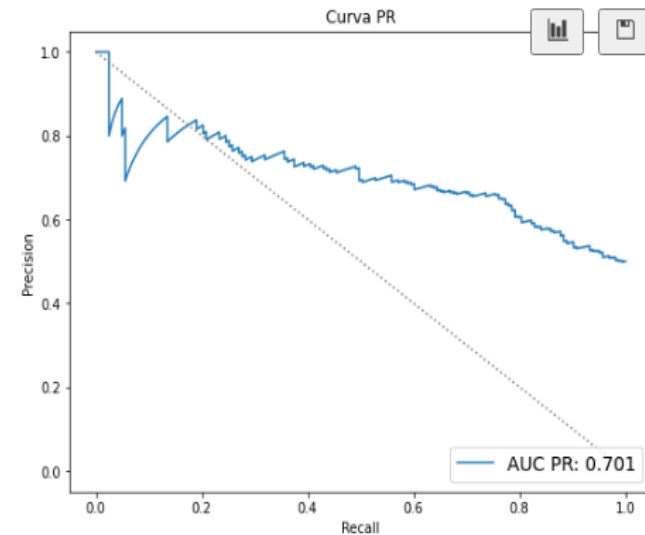
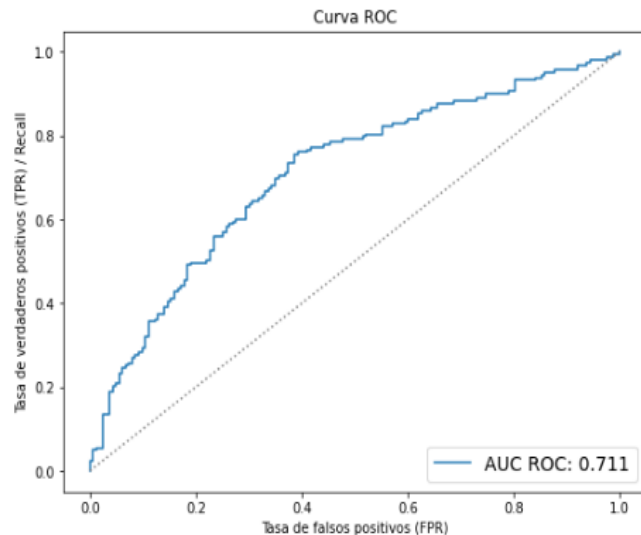
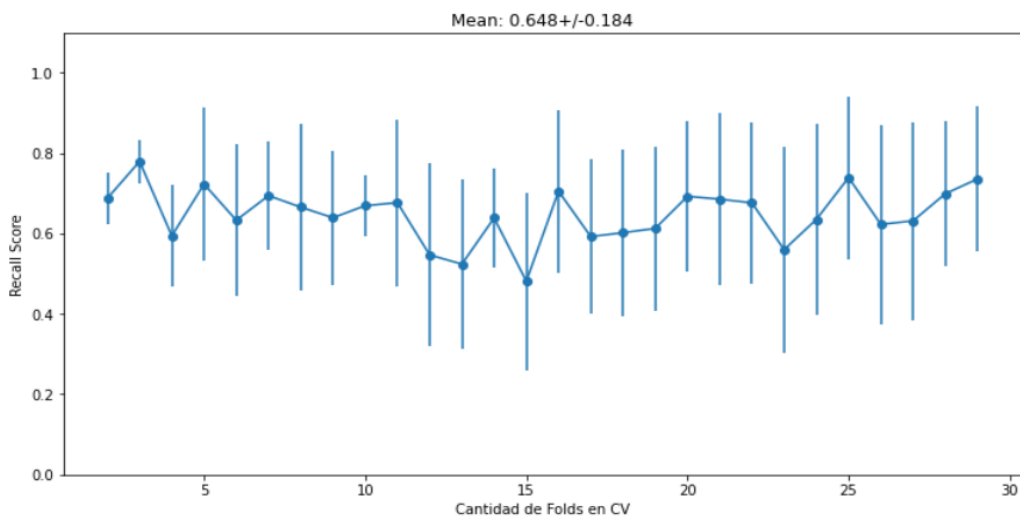
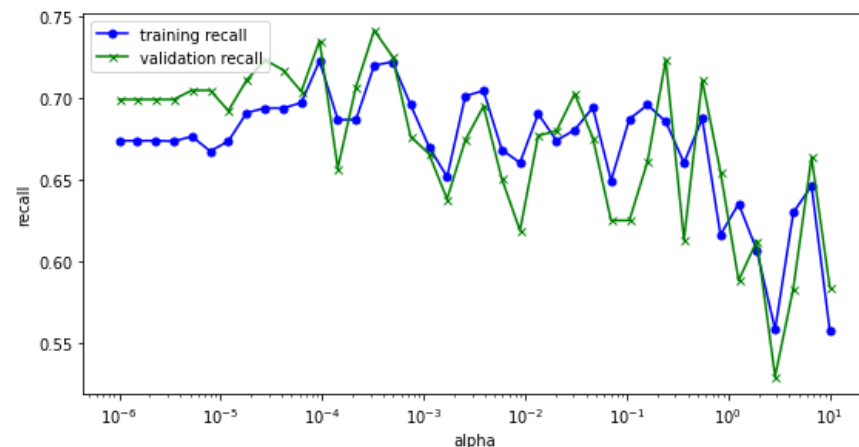
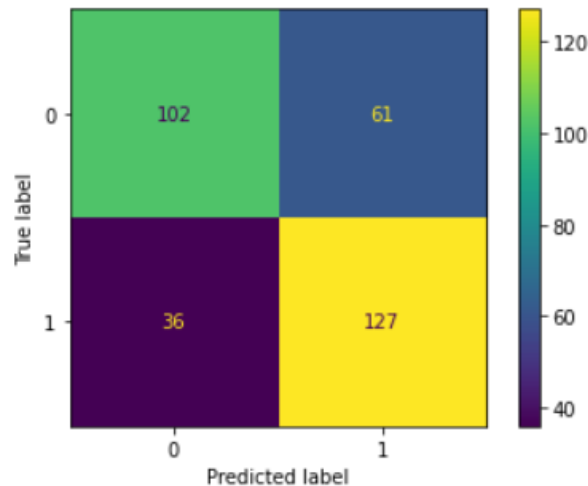
max\_iter=40000  
penalty="l2"  
alpha=1e-3  
eta0=0.04



Para optimizar utilizamos "GridSearchCV" para un trazo grueso y "validation\_curve" para ajustes finos, ambos del paquete de sklearn.model\_selection

Cross Validation Score: 0.669 +/- 0.076

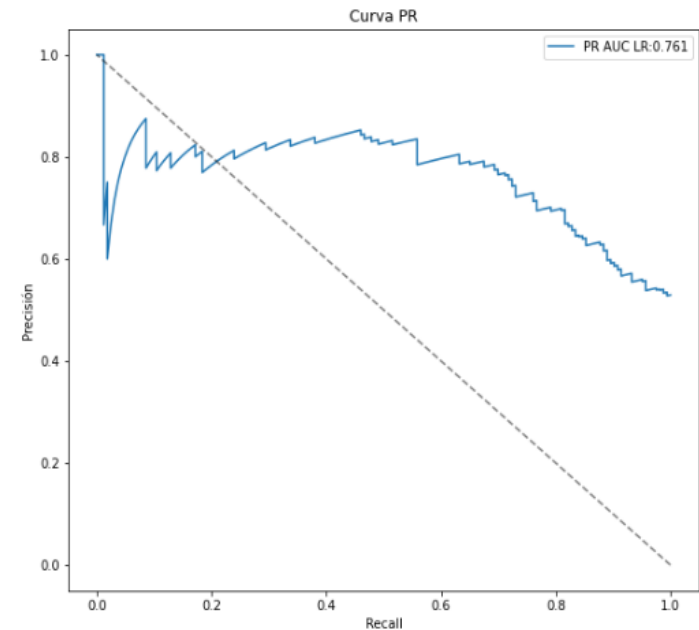
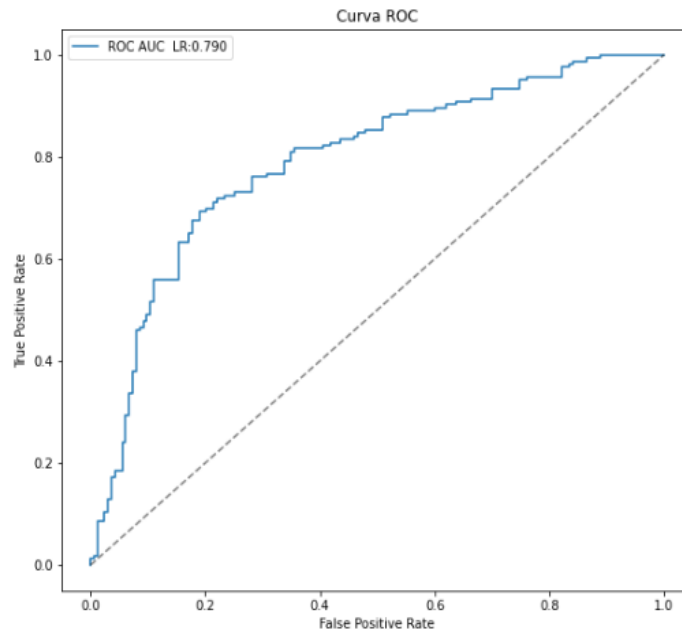
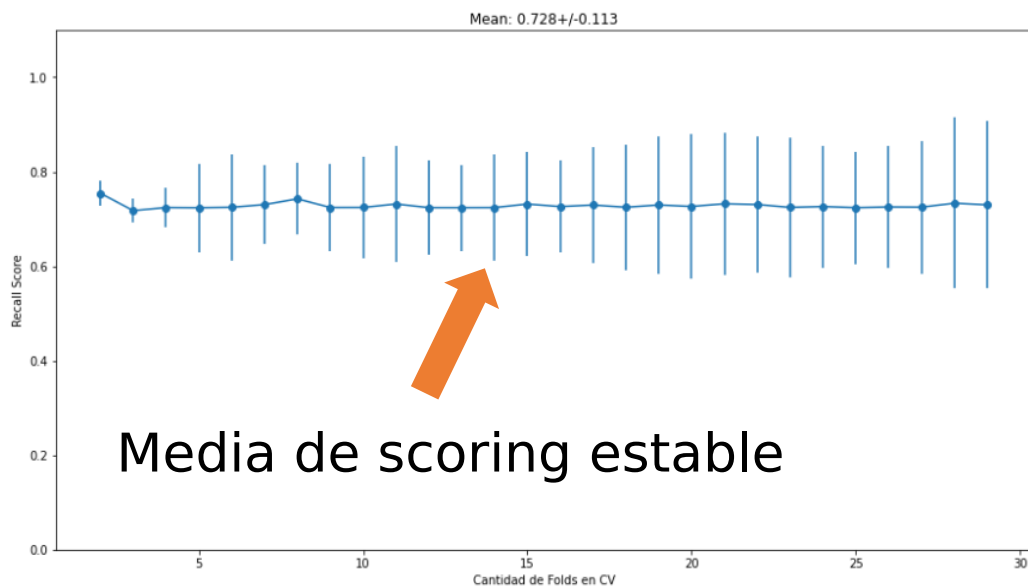
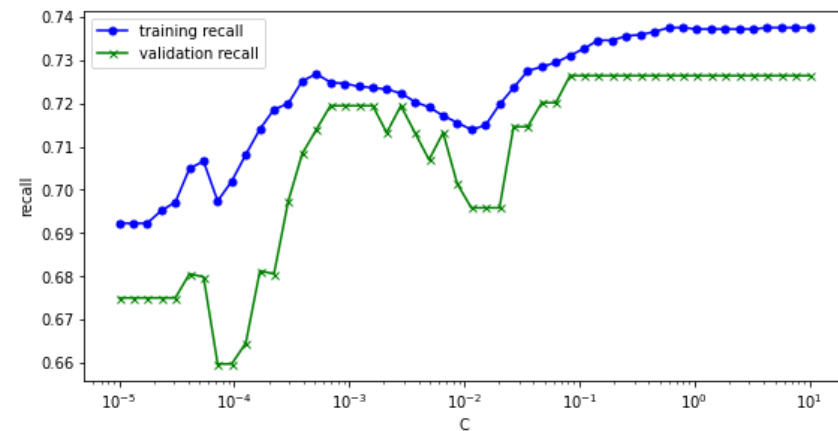
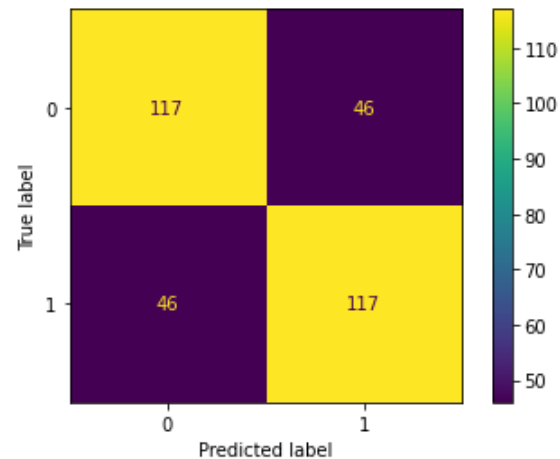
	precision	recall	f1-score	support
0	0.74	0.63	0.68	163
1	0.68	0.78	0.72	163
accuracy			0.70	326
macro avg	0.71	0.70	0.70	326
weighted avg	0.71	0.70	0.70	326



# Logistic Regression

Cross Validation Score: 0.724 +/- 0.108

	precision	recall	f1-score	support
0	0.72	0.72	0.72	163
1	0.72	0.72	0.72	163
accuracy			0.72	326
macro avg	0.72	0.72	0.72	326
weighted avg	0.72	0.72	0.72	326



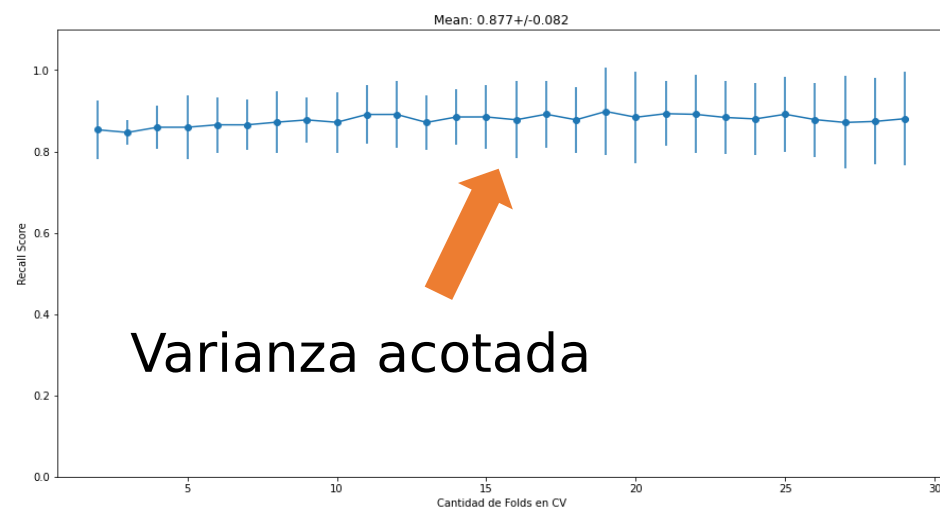
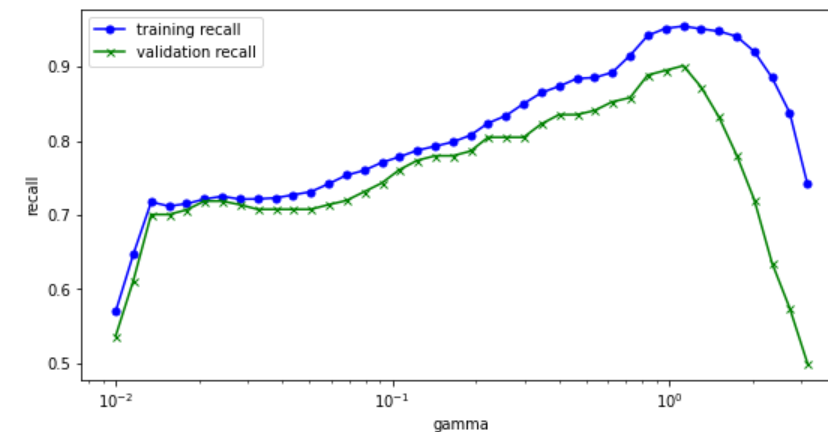
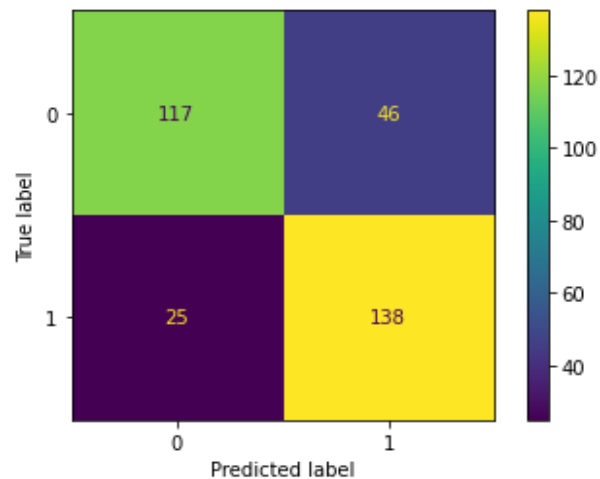


# SVC

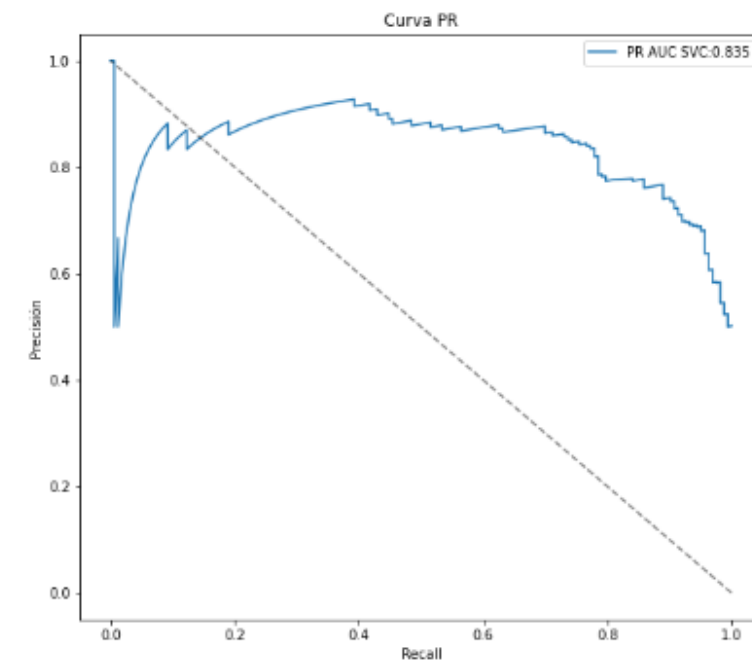
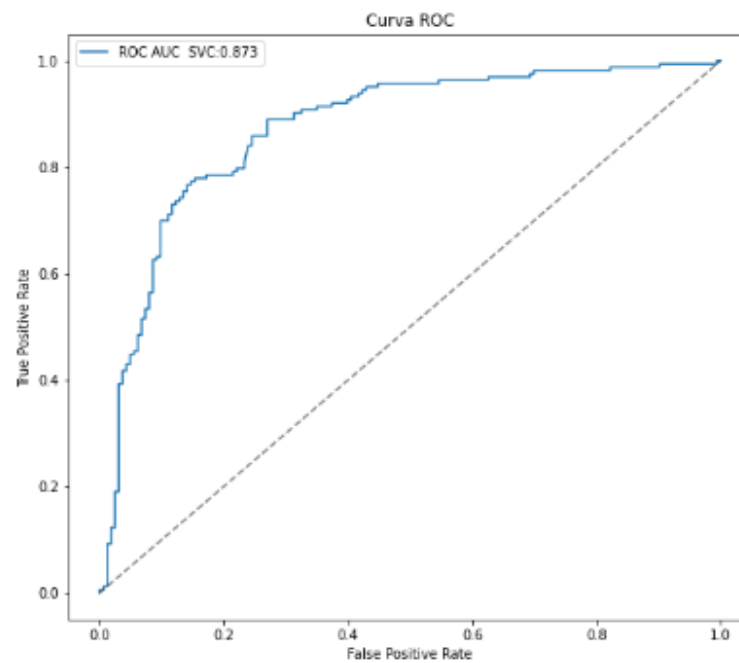
gamma=1.17  
C=0.15

Cross Validation Score: 0.872 +/- 0.075

	precision	recall	f1-score	support
0	0.82	0.72	0.77	163
1	0.75	0.85	0.80	163
accuracy			0.78	326
macro avg	0.79	0.78	0.78	326
weighted avg	0.79	0.78	0.78	326



Varianza acotada

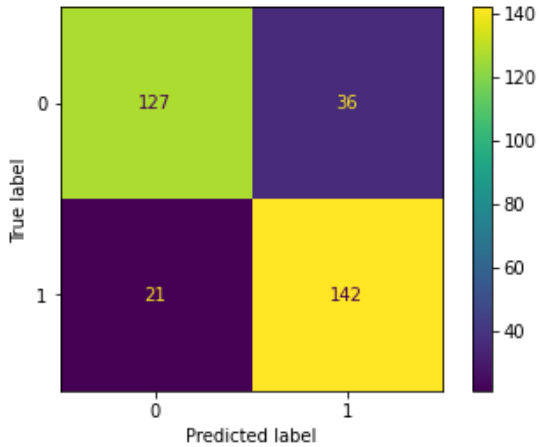


# Random Forest

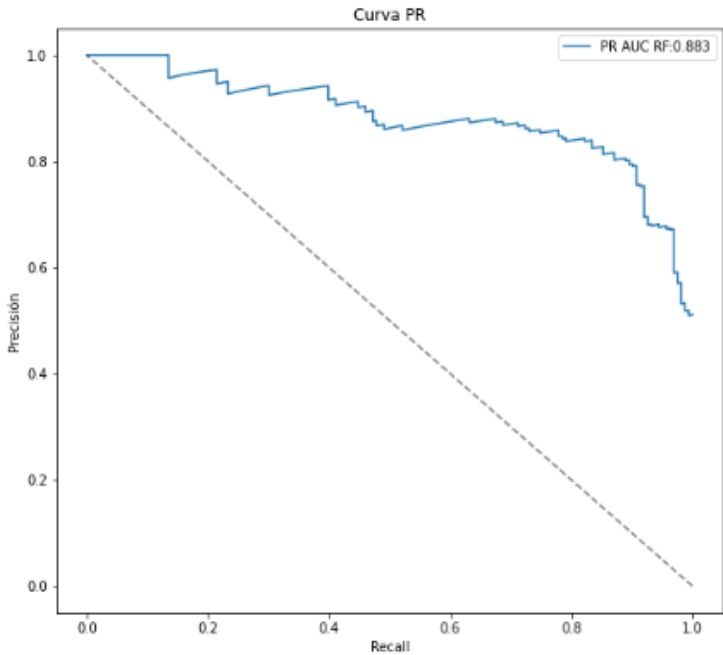
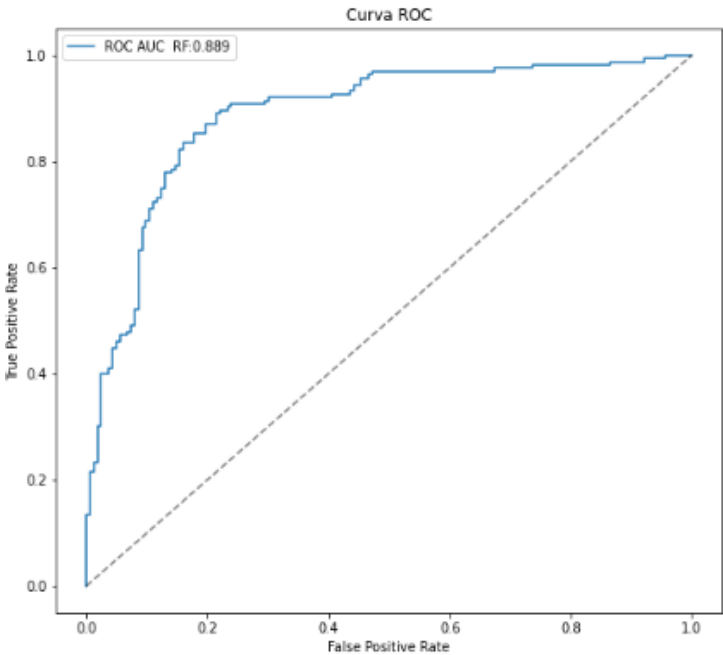
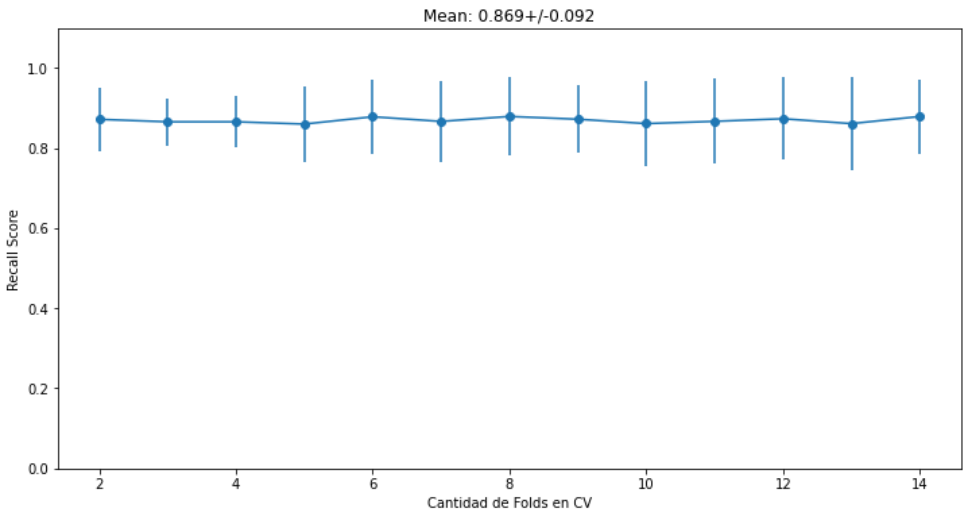
n\_estimators=400  
min\_impurity\_decrease=2e-4  
max\_depth = 8

Cross Validation Score: 0.867 +/- 0.109

	precision	recall	f1-score	support
0	0.86	0.78	0.82	163
1	0.80	0.87	0.83	163
accuracy			0.83	326
macro avg	0.83	0.83	0.82	326
weighted avg	0.83	0.83	0.82	326



La hiperparametrización en Decision Trees y Random Forest no dió resultados notorios.

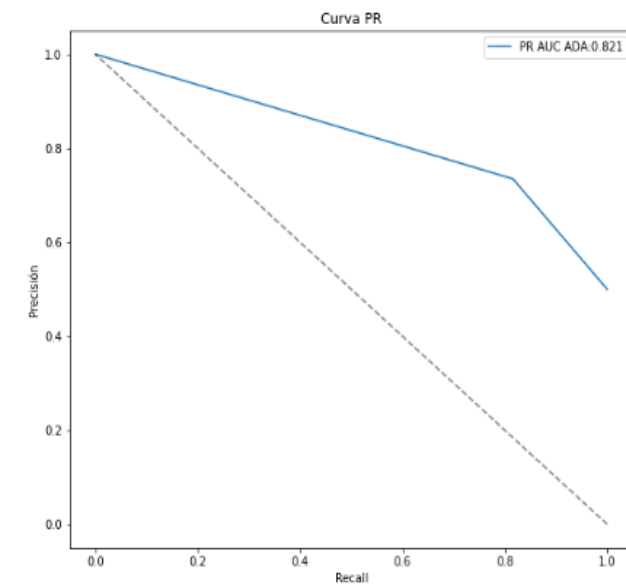
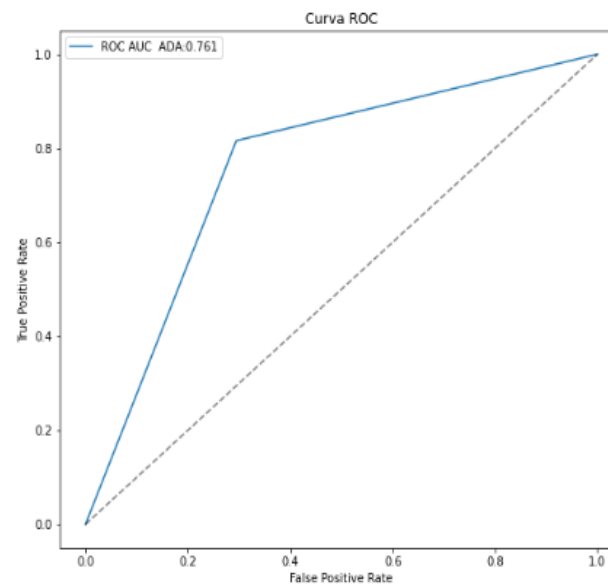
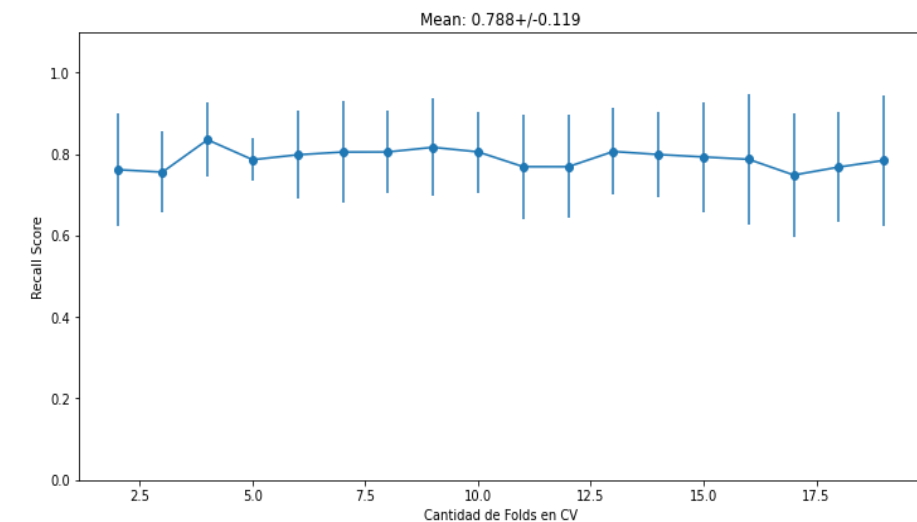
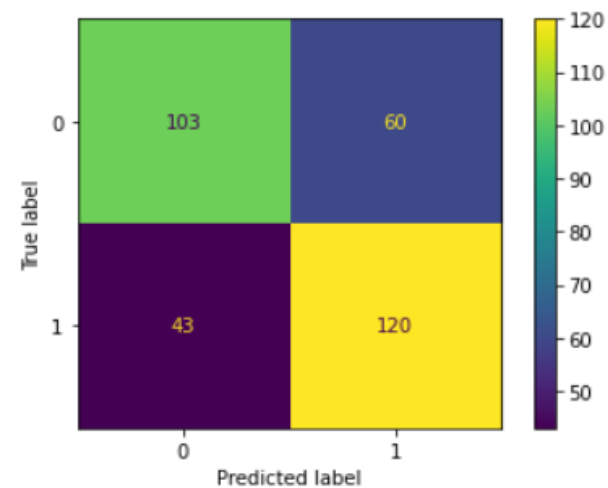


# ADA Boosting

```
base_estimator=DecisionTreeClassifier()  
learning_rate=0.2
```

Cross Validation Score: 0.811 +/- 0.077

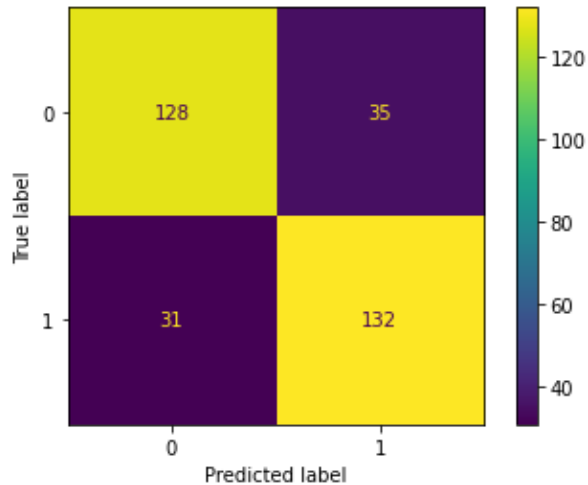
	precision	recall	f1-score	support
0	0.71	0.63	0.67	163
1	0.67	0.74	0.70	163
accuracy			0.68	326
macro avg	0.69	0.68	0.68	326
weighted avg	0.69	0.68	0.68	326



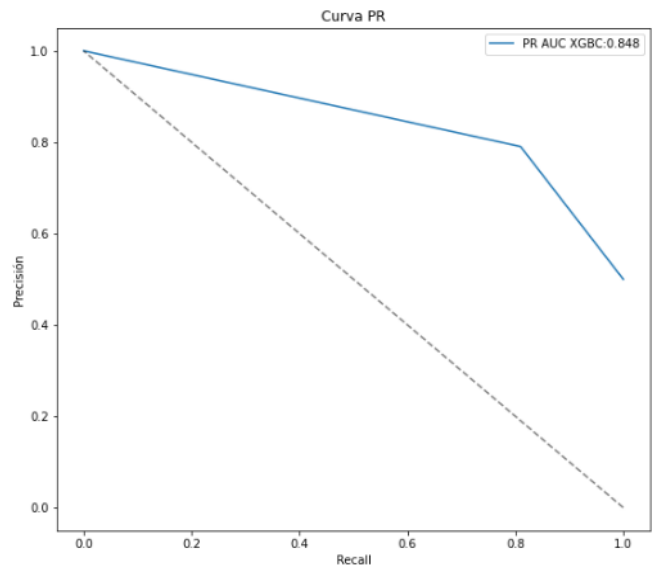
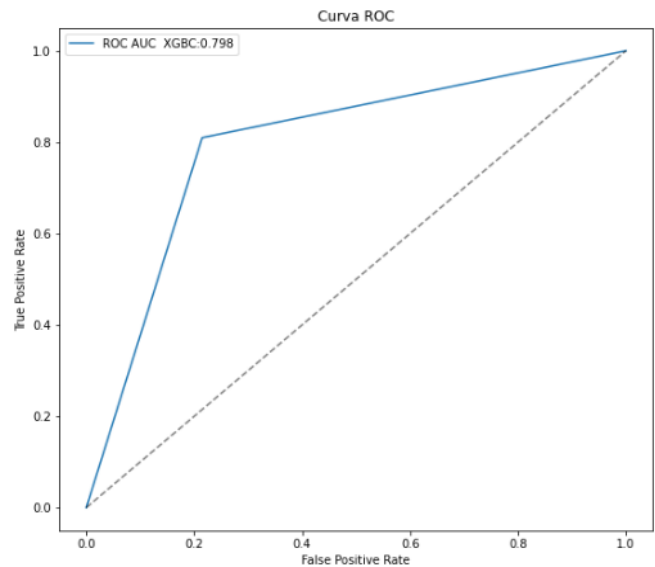
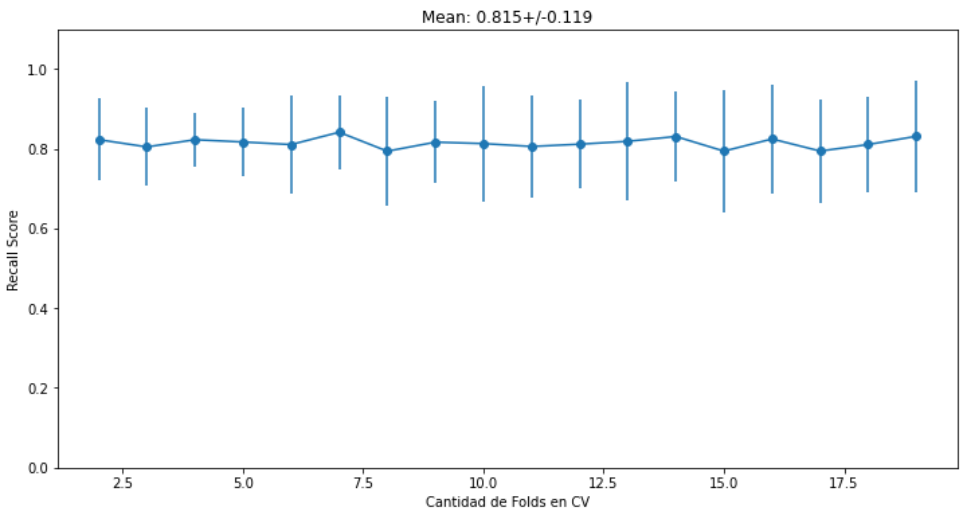
# Extreme Gradient Boosting

use\_label\_encoder=False  
objective='binary:hinge'  
learning\_rate=0.05  
n\_estimators=500

	precision	recall	f1-score	support
0	0.81	0.79	0.80	163
1	0.79	0.81	0.80	163
accuracy			0.80	326
macro avg	0.80	0.80	0.80	326
weighted avg	0.80	0.80	0.80	326



**El algoritmo dio buenos resultados considerando que necesita datos para validación y los splits en nuestro dataset son muy costosos.**

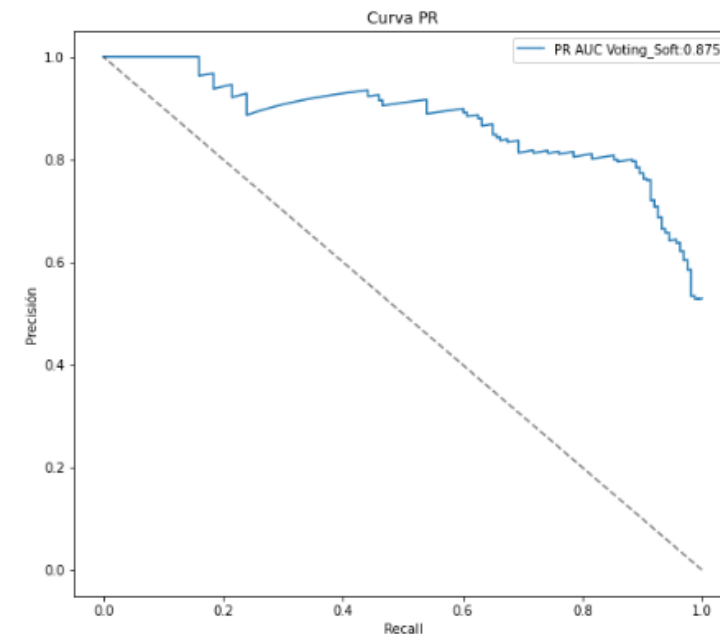
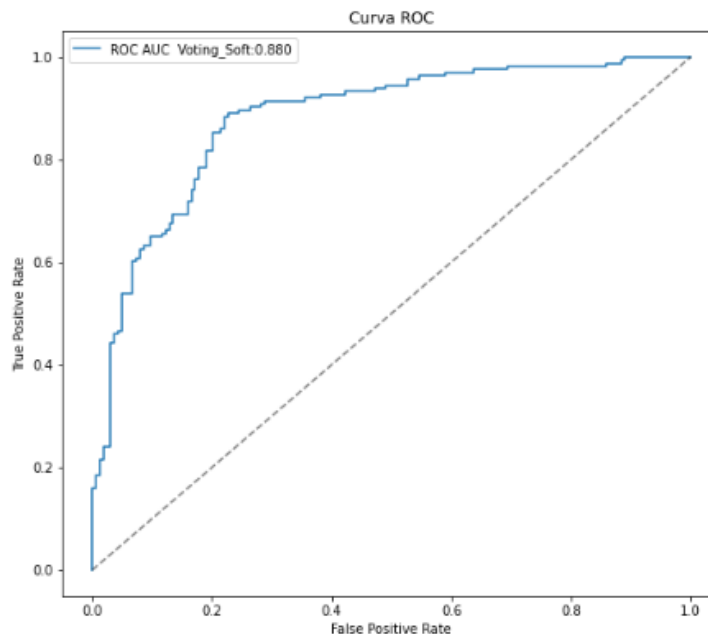
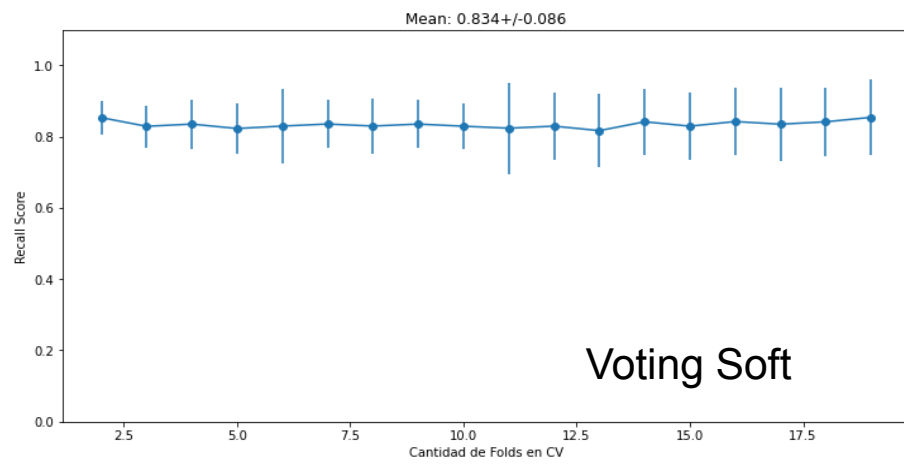
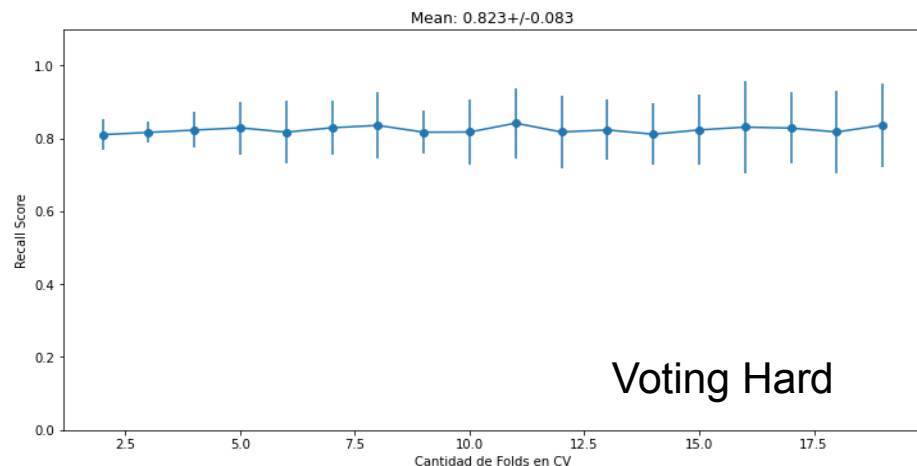
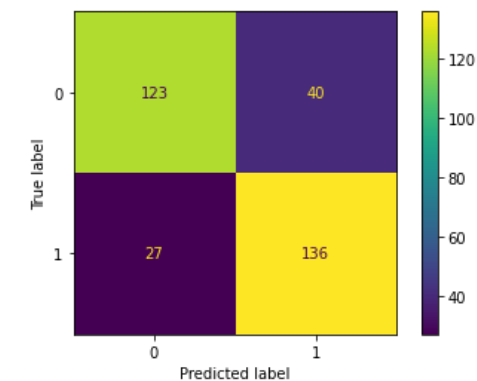
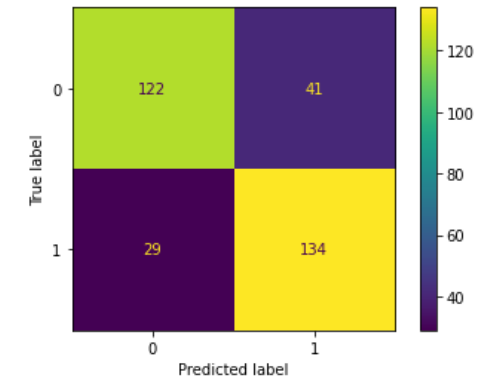


# Voting Soft y Hard

Usamos los 3 algoritmos parametrizados que consideramos mejores: RF, LR y SVC

Cross Validation Score: 0.829 +/- 0.083				
	precision	recall	f1-score	support
0	0.81	0.75	0.78	163
1	0.77	0.82	0.79	163
accuracy			0.79	326
macro avg	0.79	0.79	0.78	326
weighted avg	0.79	0.79	0.78	326

Cross Validation Score: 0.830 +/- 0.082				
	precision	recall	f1-score	support
0	0.82	0.75	0.79	163
1	0.77	0.83	0.80	163
accuracy			0.79	326
macro avg	0.80	0.79	0.79	326
weighted avg	0.80	0.79	0.79	326



# Neuronal Network

Loss: Binary Cross-Entropy  
Optimizer: Adam (lr = 1e-4)  
Metrics: Recall  
Epochs: 600  
Batch = 32  
Earling Stopping

```
input = keras.layers.Input(shape=[6,])
flatten = keras.layers.Flatten()(input)
hidden1 = keras.layers.Dense(100, activation="relu")(flatten)
dropout = keras.layers.Dropout(rate=0.2)(hidden1)
hidden2 = keras.layers.Dense(100, activation="relu")(hidden1)
output = keras.layers.Dense(1, activation="sigmoid")(hidden2)

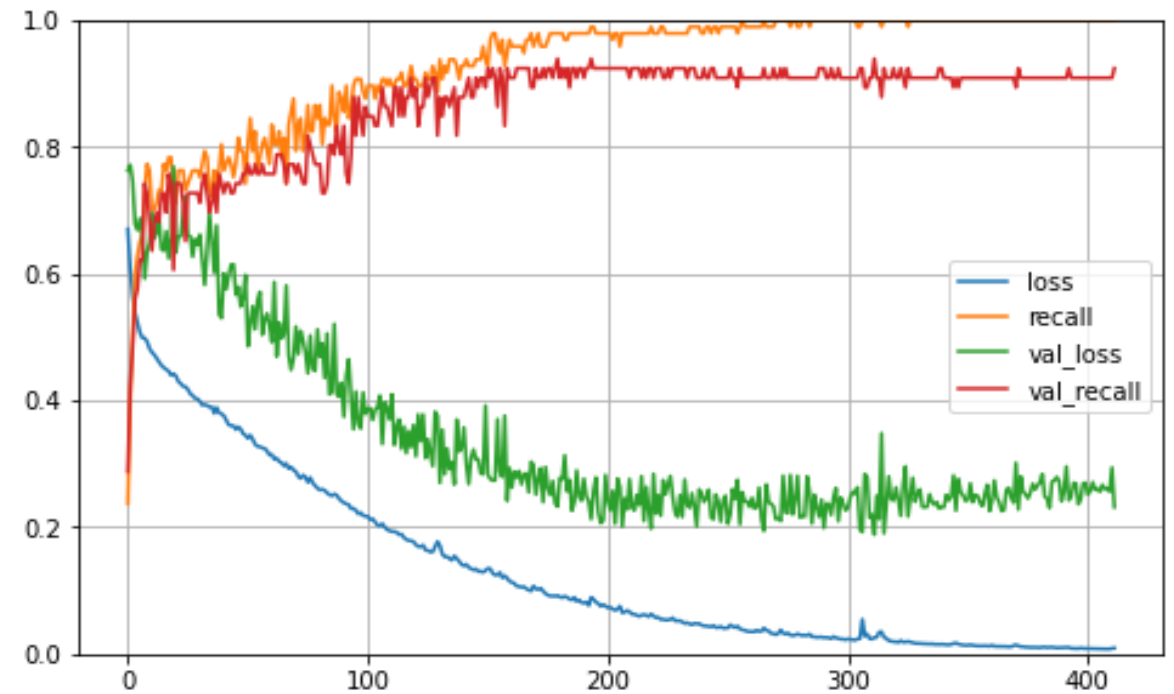
model = keras.models.Model(inputs=[input], outputs=[output])
```

Model: "model"

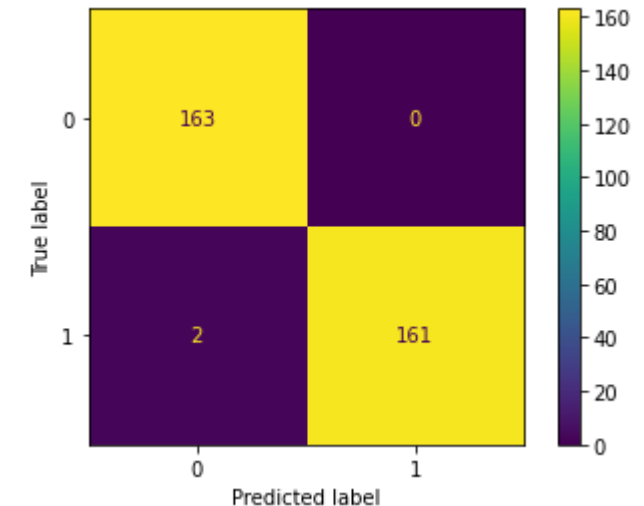
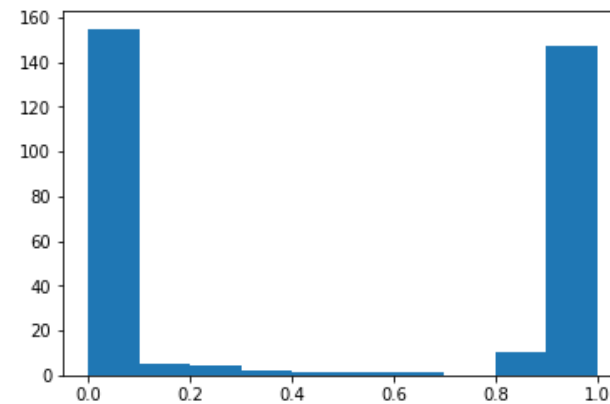
Layer (type)	Output Shape	Param #
=====		
input_1 (InputLayer)	[(None, 6)]	0
flatten (Flatten)	(None, 6)	0
dense (Dense)	(None, 100)	700
dense_1 (Dense)	(None, 100)	10100
dense_2 (Dense)	(None, 1)	101
=====		
Total params: 10,901		
Trainable params: 10,901		
Non-trainable params: 0		

Epoch 463/600

9/9 [=====] - 0s 2ms/step - loss: 0.0067 - recall: 1.0000 - val\_loss: 0.1678 - val\_recall: 0.9394



```
plt.hist(model.predict(feats_train))
```



# En resumen...

- Las hiperparametrización logró mejorar el score de entrenamiento en casi todos los casos.
- Hay una caída sustancial en los scorings de testing. Era esperable tras observar la alta varianza en entrenamiento.
- Se observó mejores resultados en modelos con mas “bias”.
- SMOTE fue una herramienta indispensable para nuestro análisis y los scorings.

