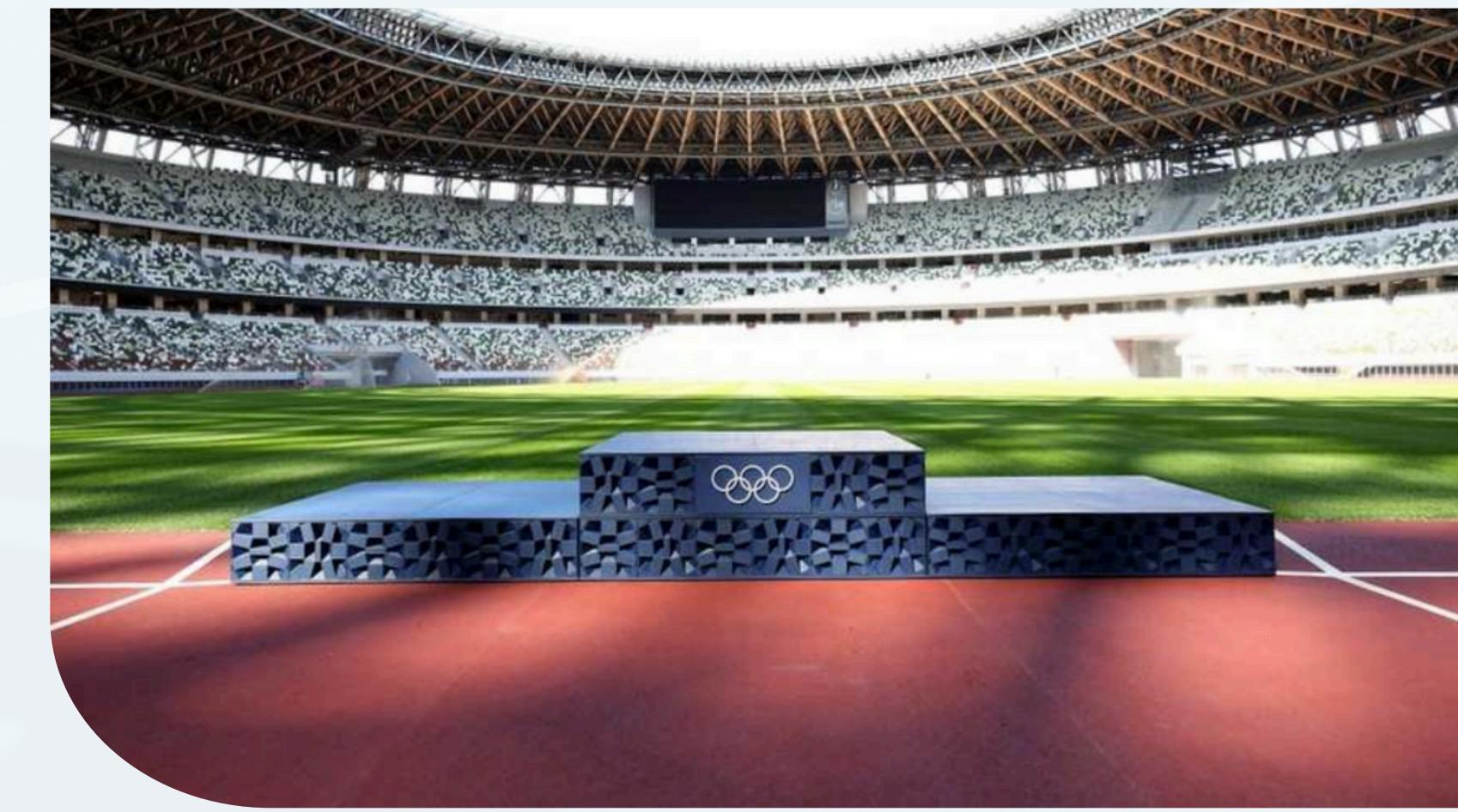




TOWARDS THE GLORY

WITH MACHINE LEARNING



**ÁLVARO
MARTINEZ
VALIENTE**

**PROYECTO DE ML SUPERVISADO
Y NO SUPERVISADO**

Contexto

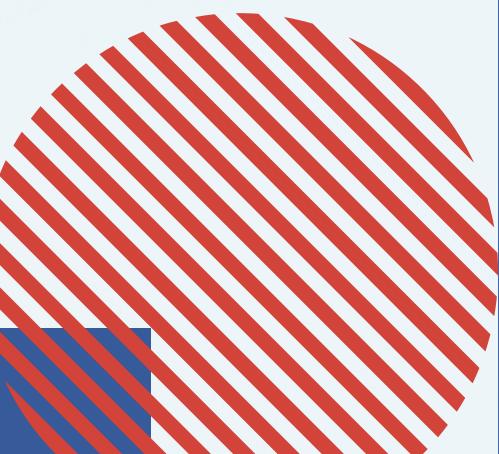
Dataset: Todos los atletas en los JJOO con y sin medalla, desde los primeros juegos olimpicos en Atenas en 1896 hasta Rio 2016.

Columnas: ID, Name, Sex, Age, Height, Weight, Team, NOC, Games, Year, Season, City, Sport, Event, Medal.

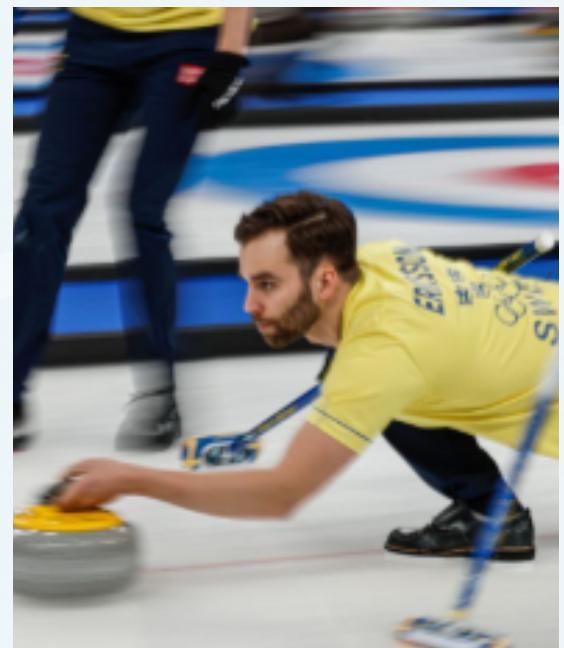
Numero de registros: 271116 filas x 15 columnas.

Tipos de variables: Categóricas y numéricas.

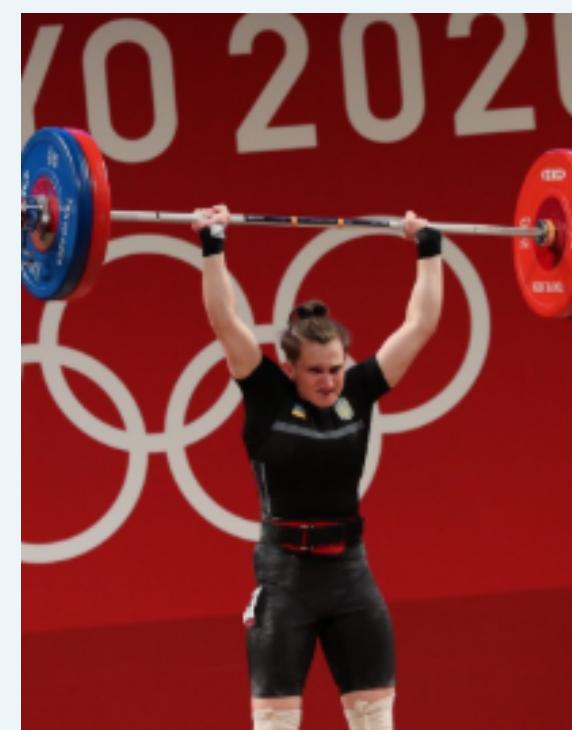
Predicción: si un deportista va a ganar una medalla en los proximos JJOO, la cantidad de medallas que ganara un país...



Estructura del proyecto



Limpieza de datos



Analisis univariante



Analisis bivariante



Modelado baseline & optimización de balanceo de target



Optimización de modelo & hiperparámetros

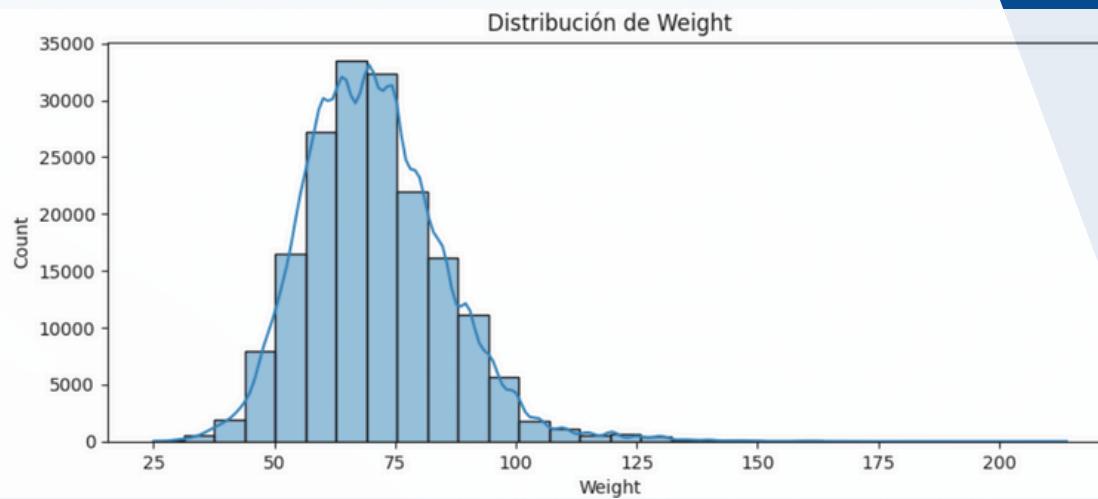
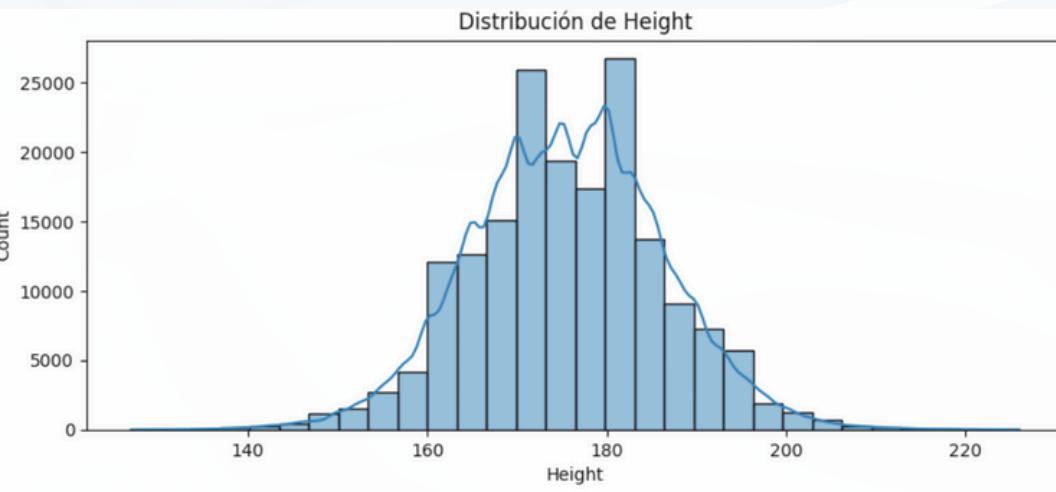
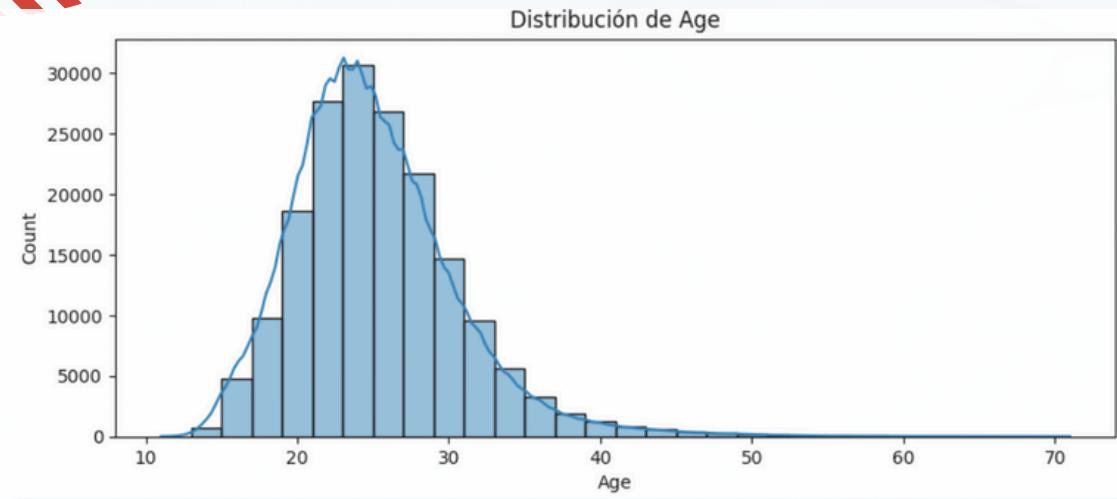


Streamlit

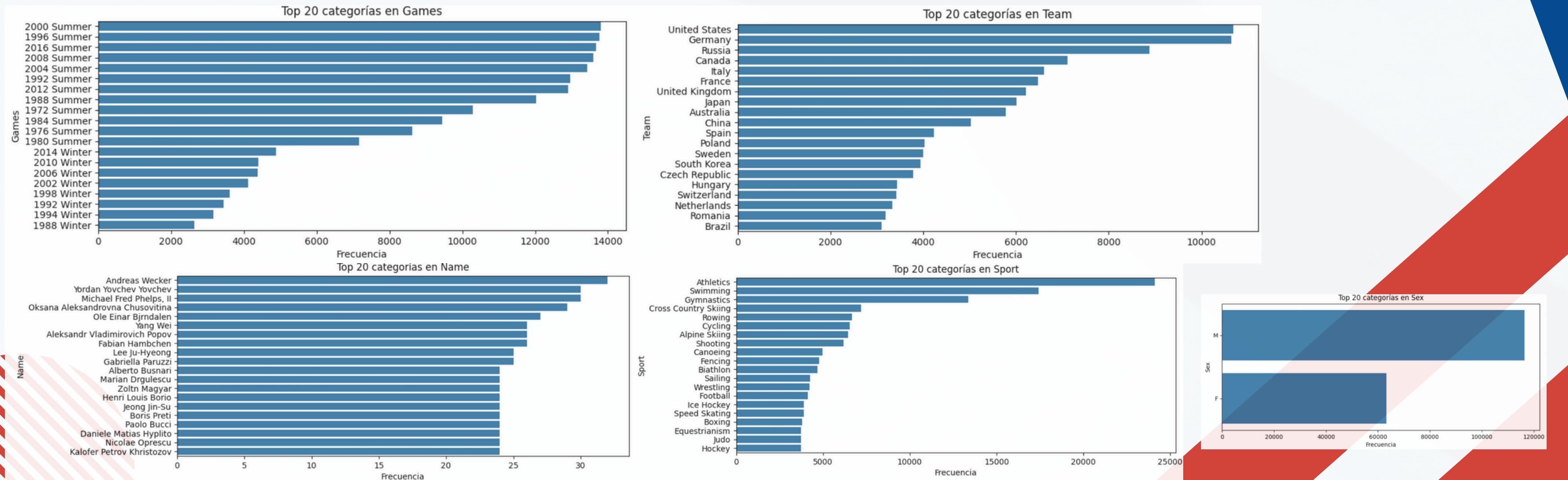
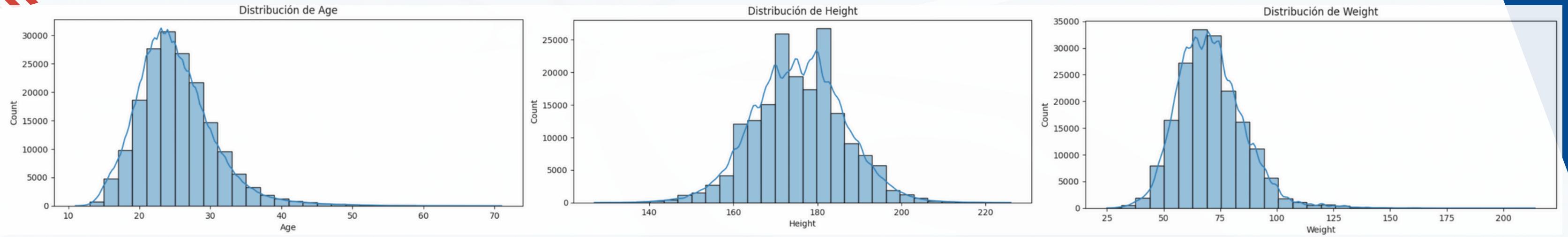
Limpieza de datos

- Cambiamos los NaN de la columna medalla por 'No Medal'
- Para eliminar los nulos de las columnas Age, Height y Weight, vamos a crear una media agrupando por pais y por deporte.
- **Año:** Eliminamos datos de antes de 1970 => Menos utiles para predecir y reducimos número de filas.
- **Team:** se llaman diferente segun el año y también existen paises nuevos, otros desaparecidos... => Asignamos el nombre oficial de paises que compiten en la actualidad y asignamos nombre de paises nuevos a paises antiguos.
Ej: 'URSS' → 'Russia' al final pasamos a 206 equipos de los 1180 al inicio.
- **Sport:** Comparamos disciplinas olimpicas en la actualidad con pasadas => Eliminamos baseball y softball que ya no son olímpicas.
- **Outliers:** eliminamos valores atípicos en edad (>60 años), en peso(> 180kg y < 30 Kg) y en altura (> 220cm y <120cm)

Analisis univariante

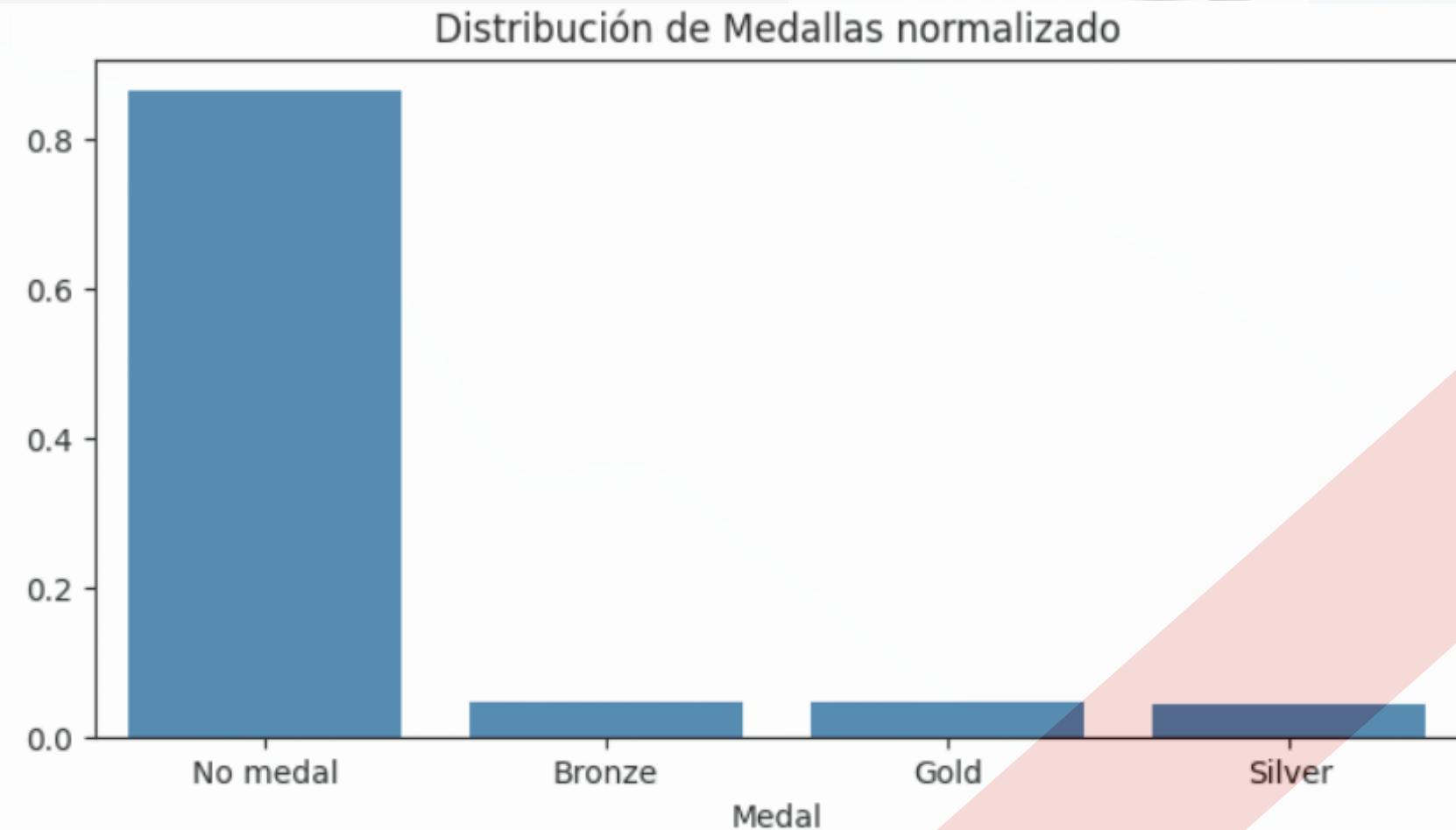
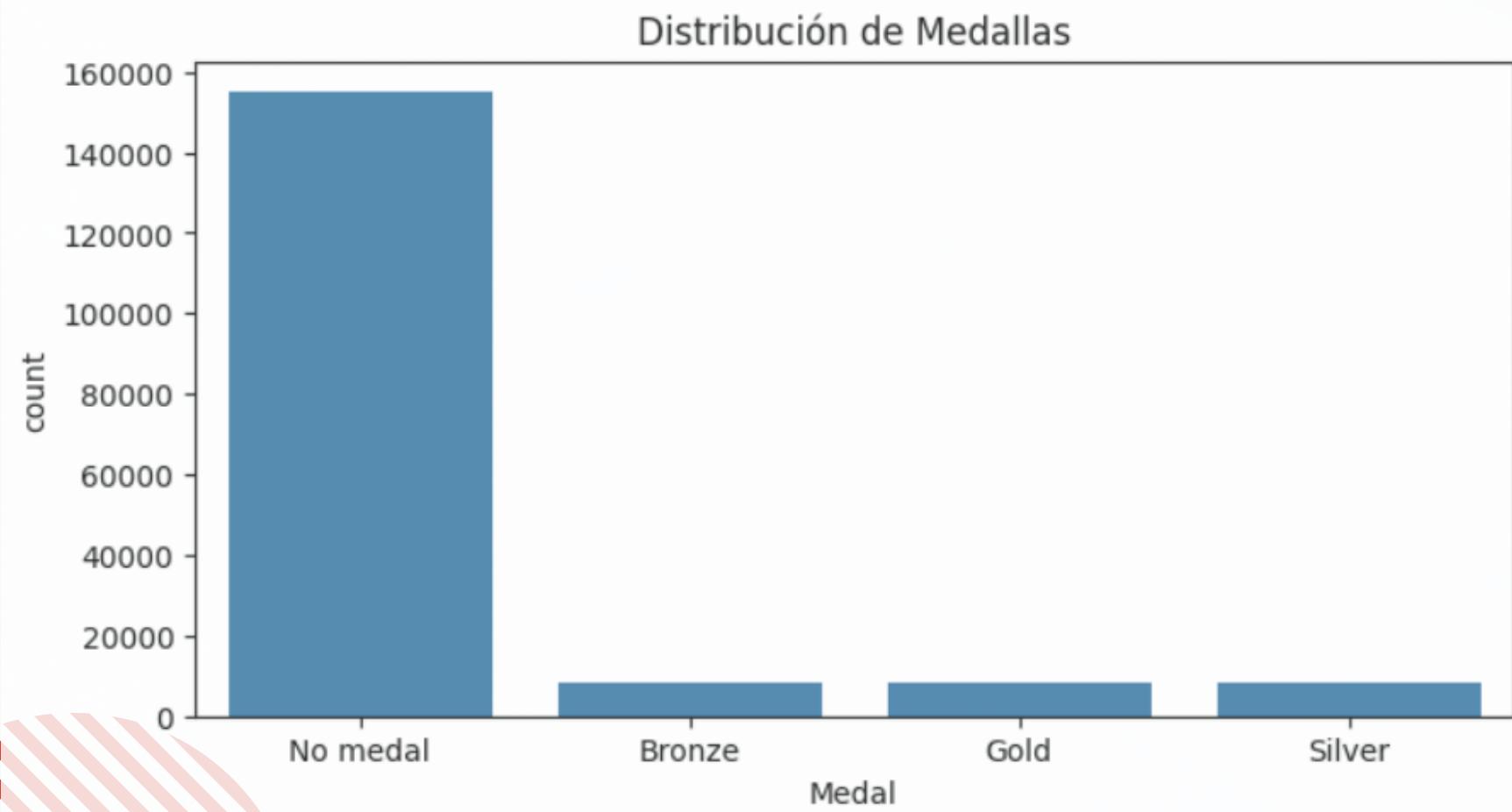


Analisis univariante

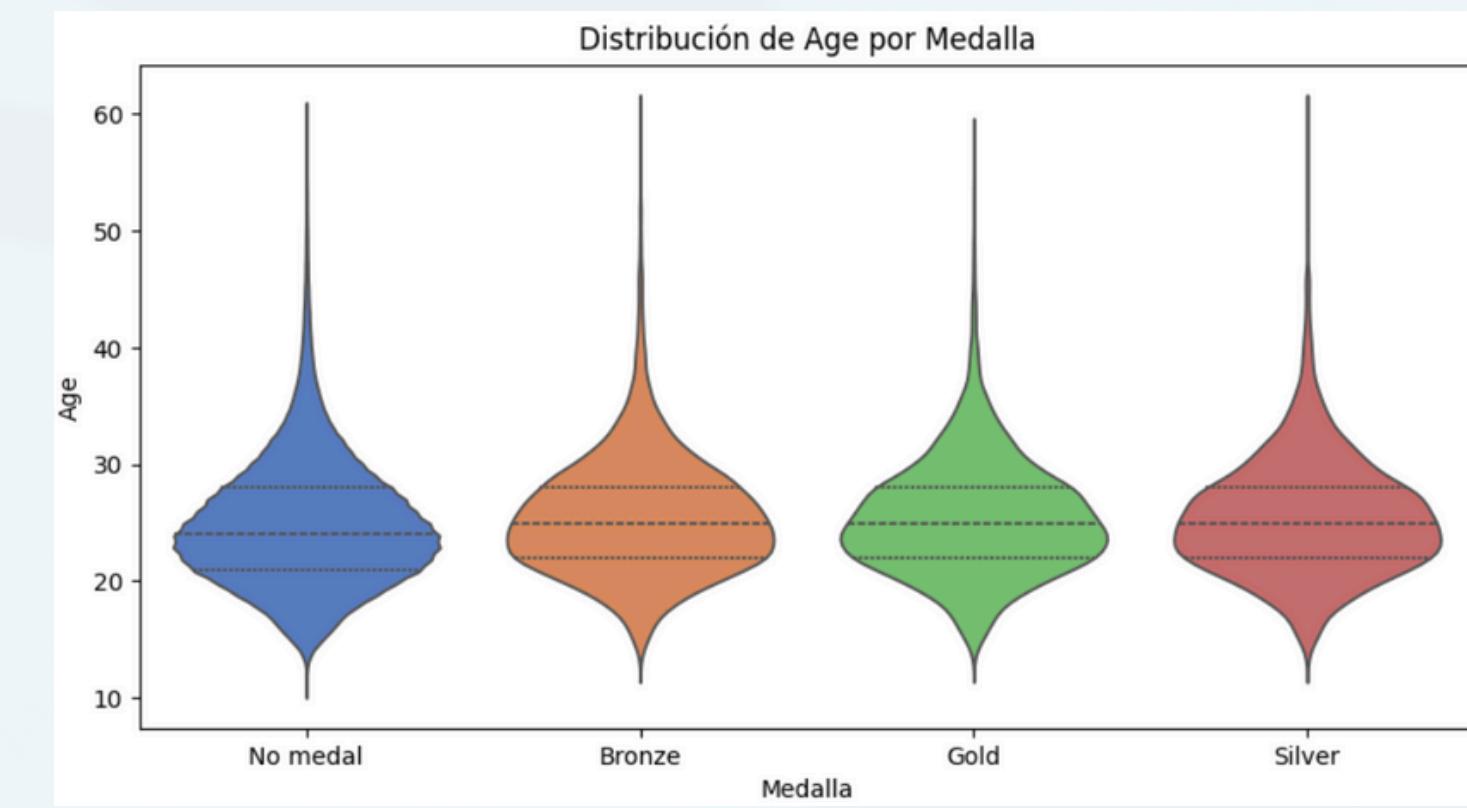
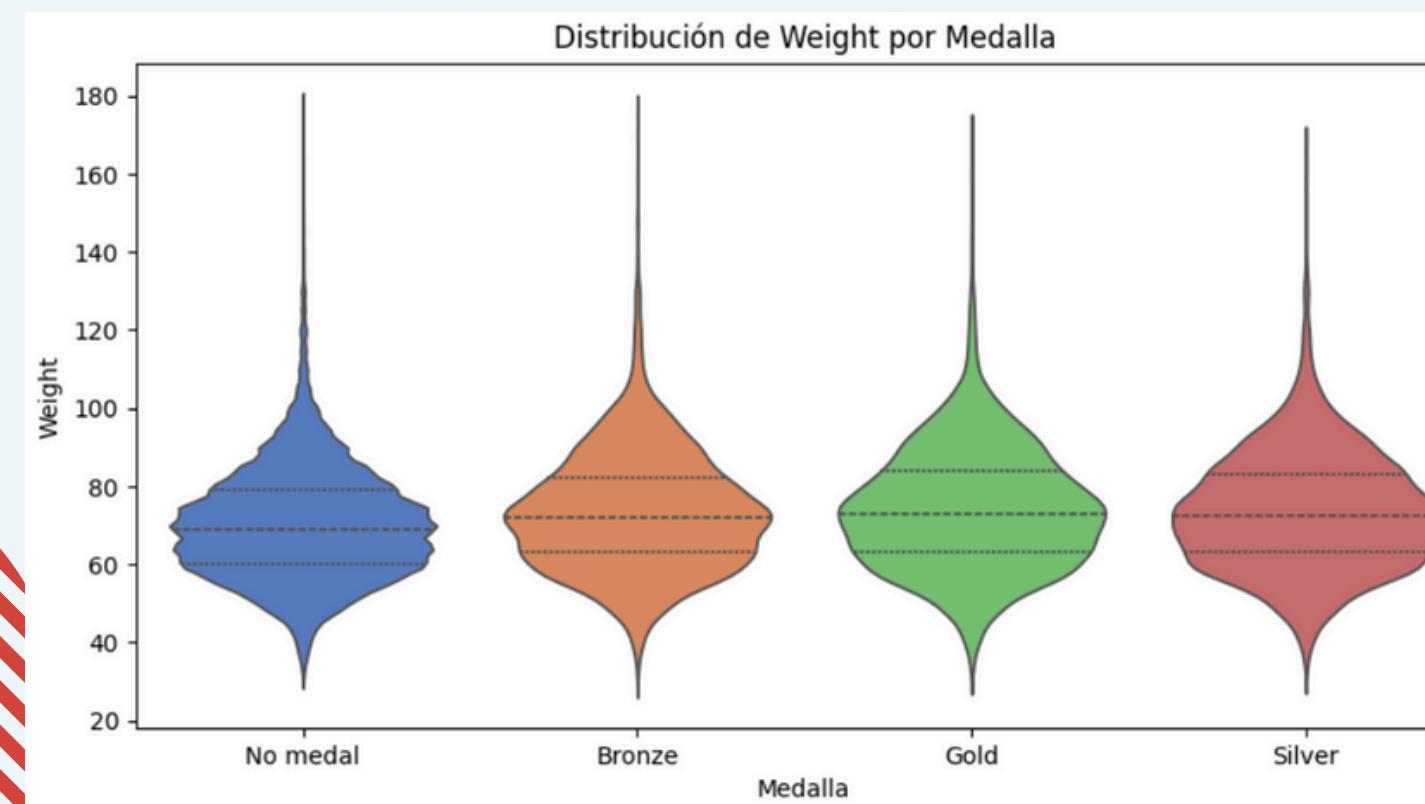
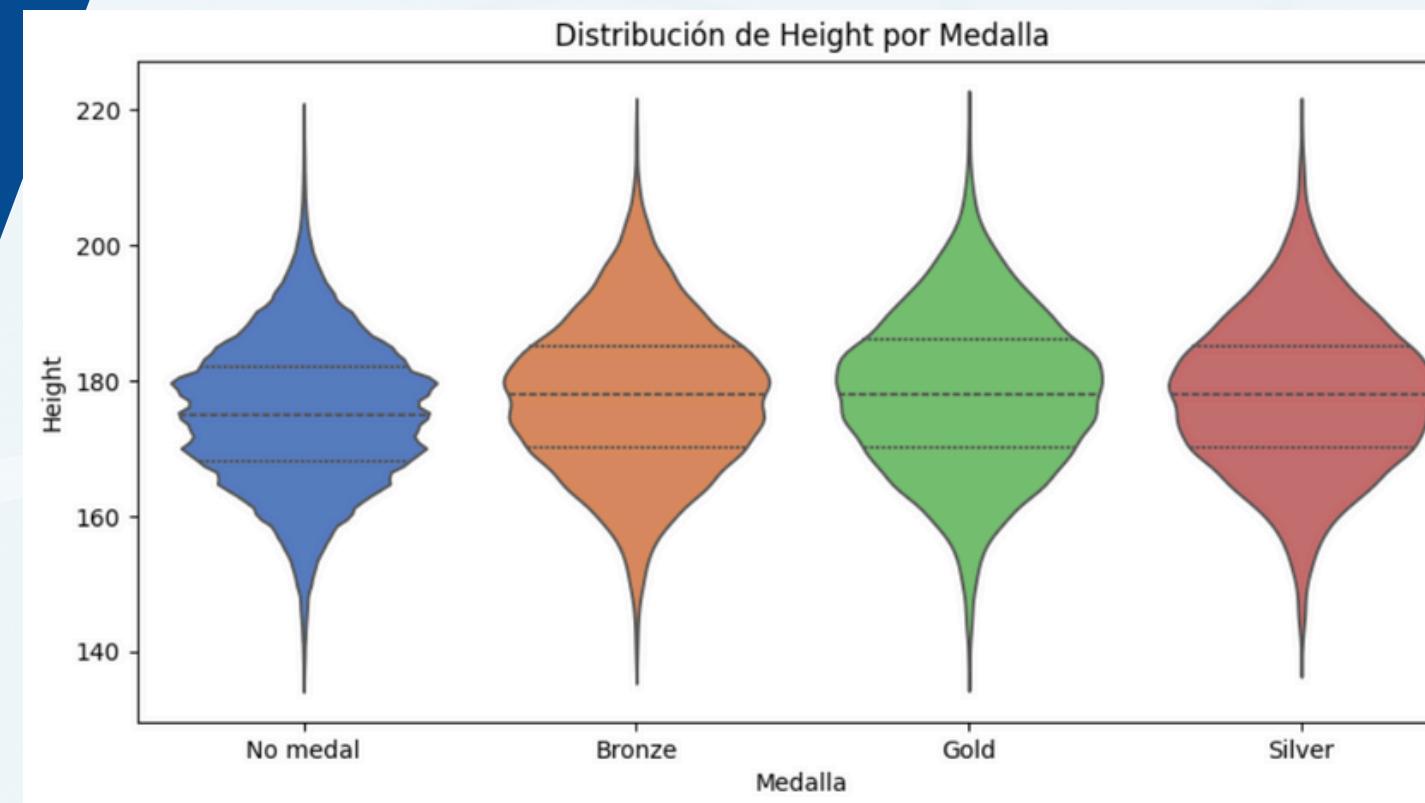


Analisis univariante

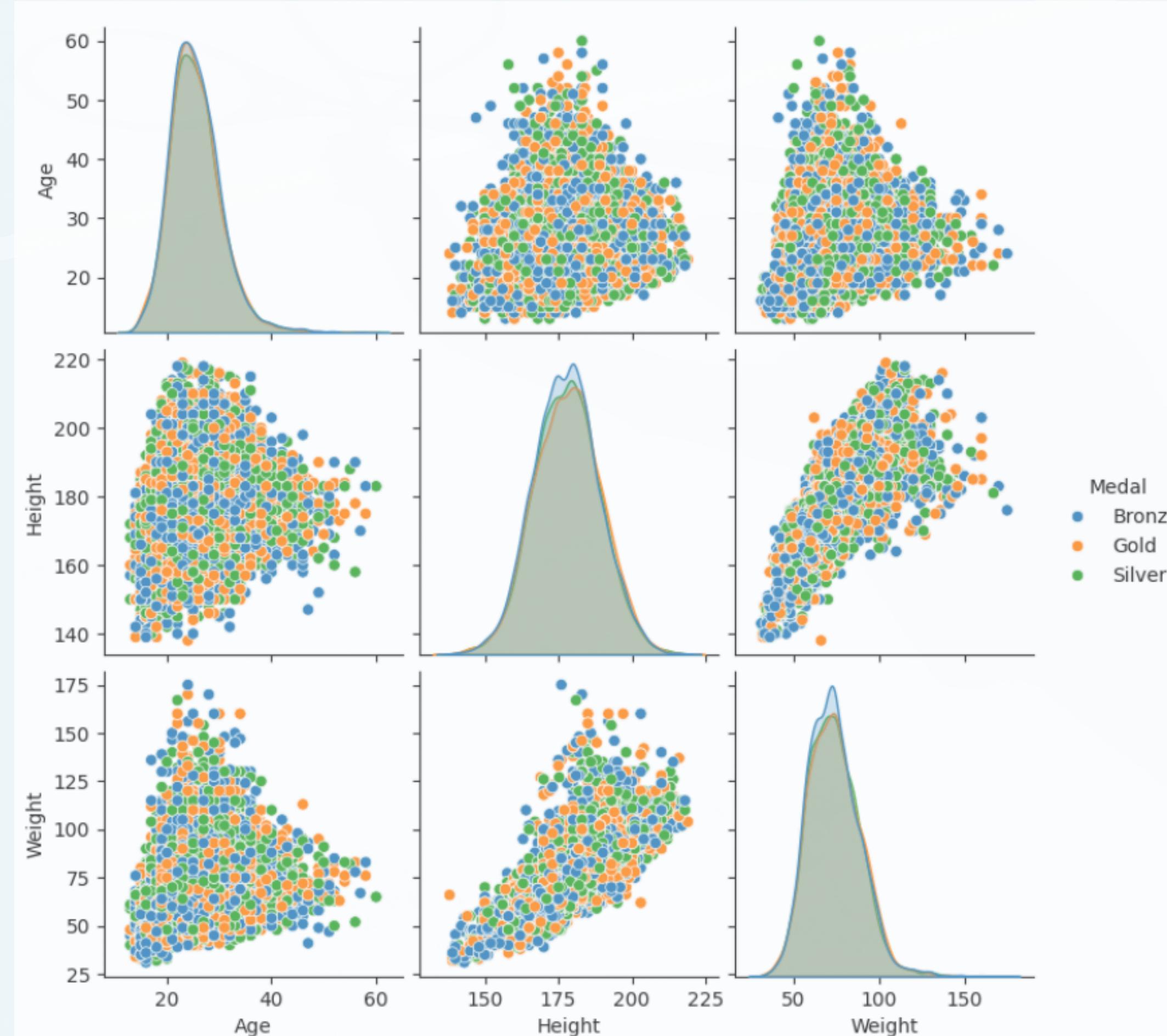
¿Alguien intuye un problema?



Analisis bivariante

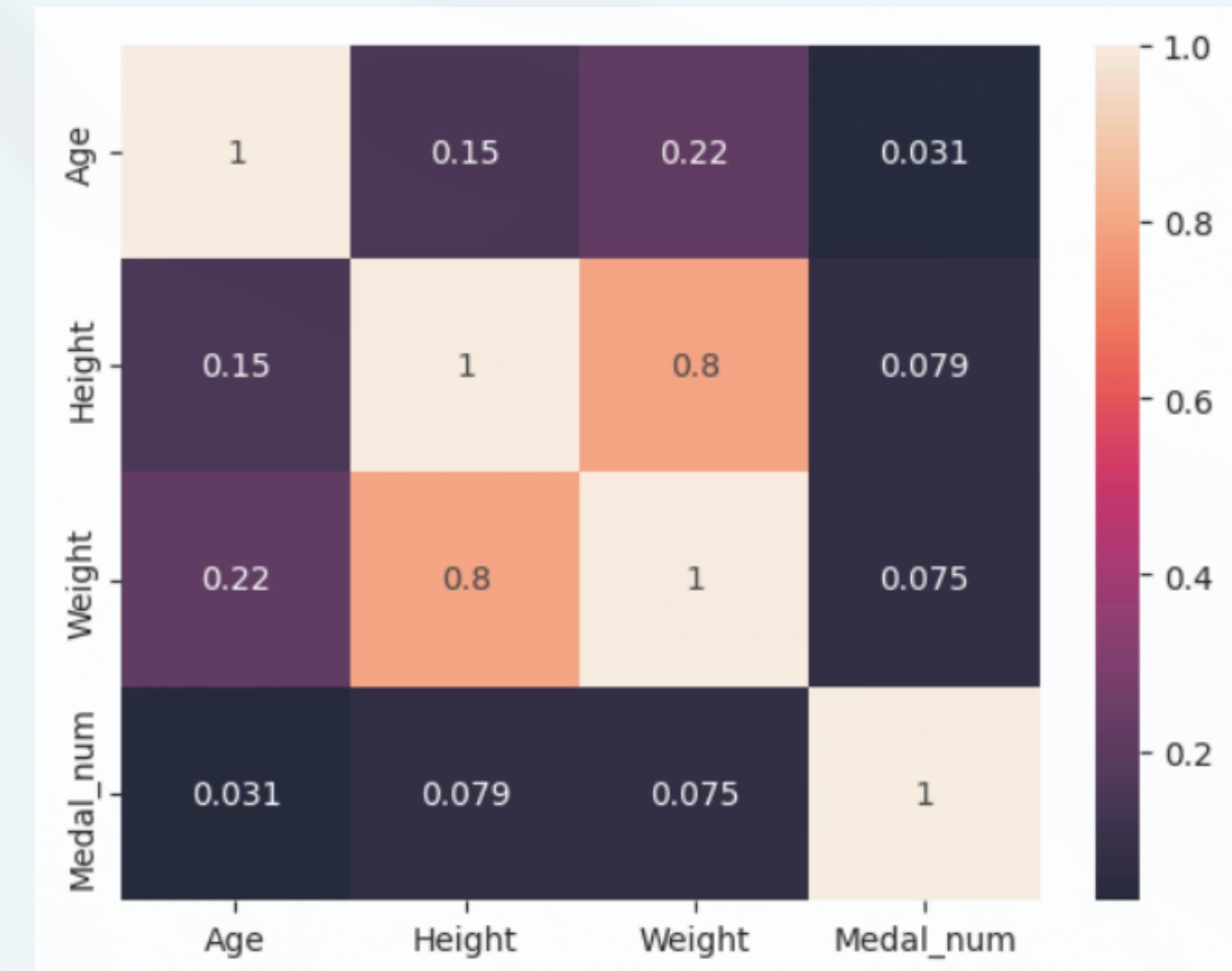
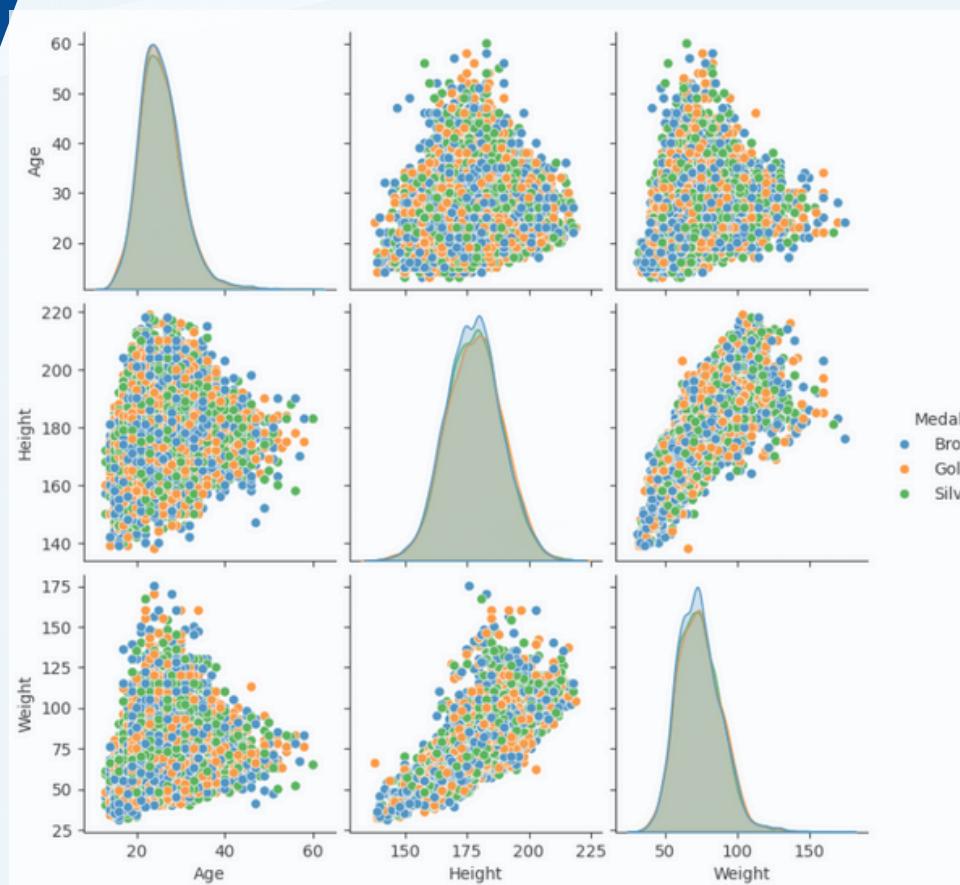


Analisis bivariante



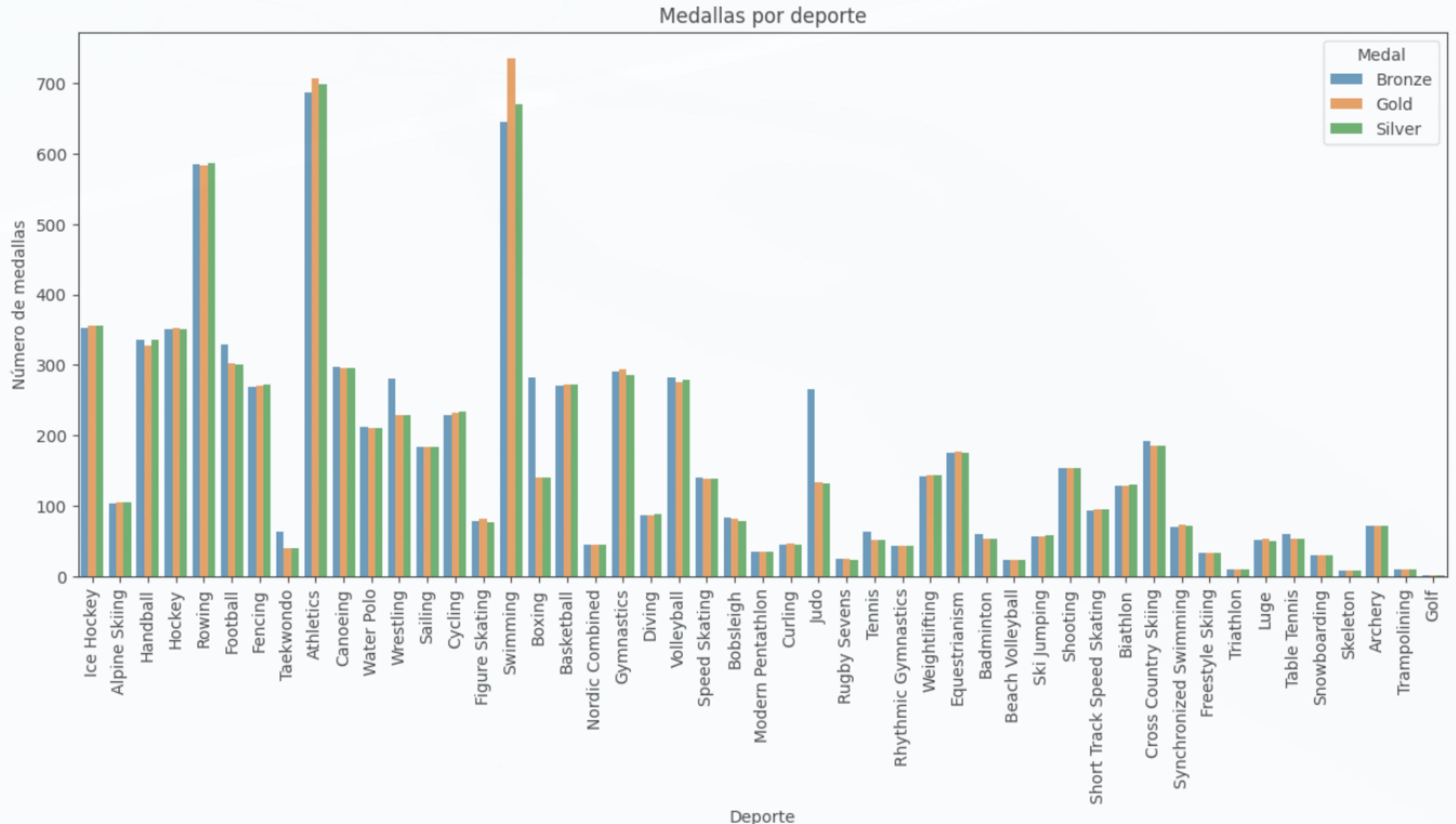
Analisis bivariante

Puntos muy mezclados y curvas superpuestas => dificil target multiclase con estas variables



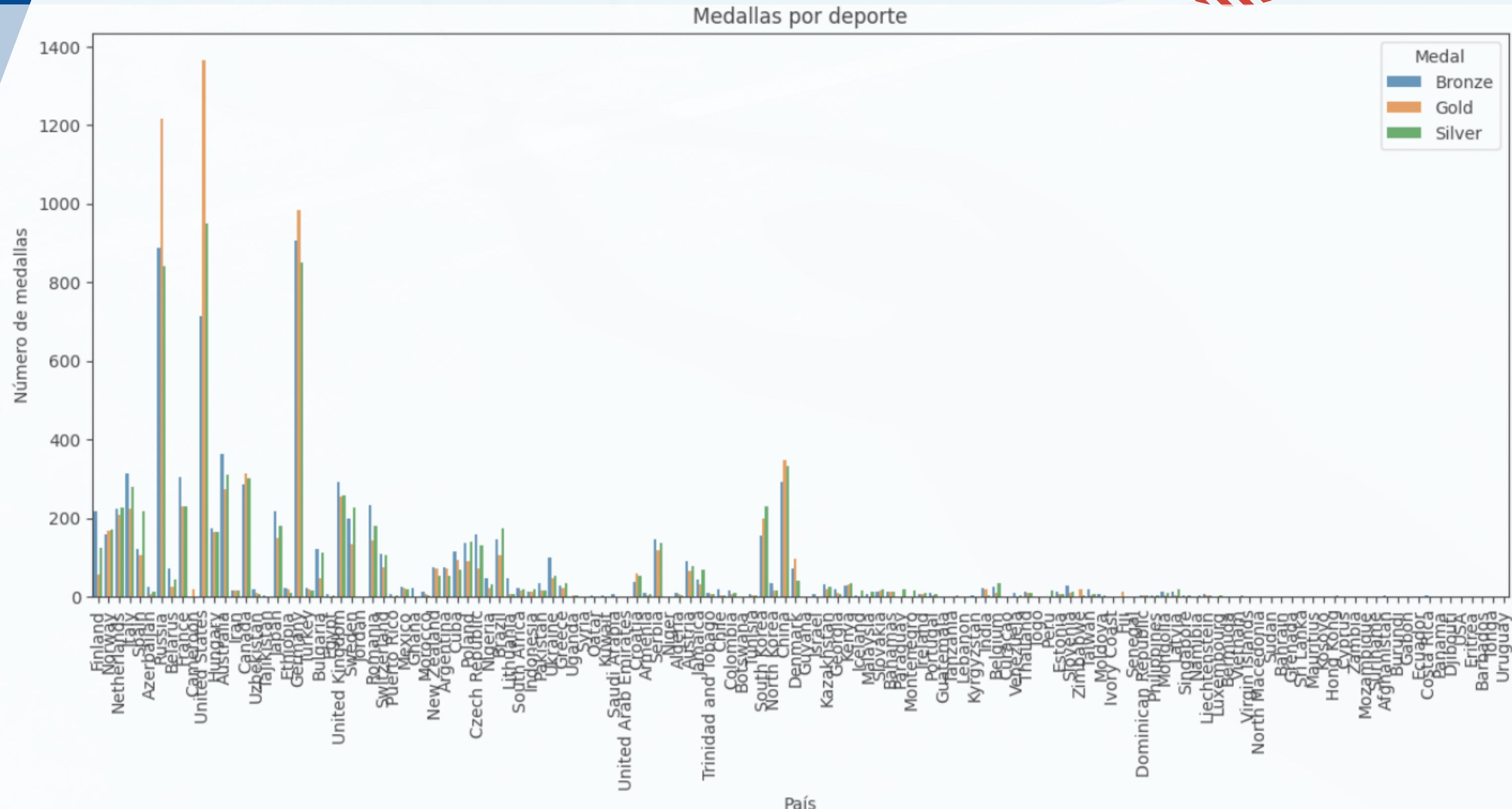
Analisis target multiclases y target binario (Medalla/No medalla)

Analisis bivariante



Analisis bivariante

Medallas por deporte



Modelado baseline & optimización de balanceo de target

Debido al gran desbalanceo de clases, la optimización del baseline se basa en elegir la mejor técnica del balanceo más que en encontrar el mejor modelo (que sera la etapa siguiente)

Las técnicas utilizadas son: SMOTE, SMOTETomek, Undersampling, SMOTE + Undersampling

Preprocessing ==> Para variables numericas utilizamos un Standard Scaler, para las variables Categoricas, usamos un Target Encoder

Creamos un pipeline de base usando un modelo de Random Forest con:

- n_estimators=300,
- max_depth=6,
- min_samples_leaf = 10,
- class_weight="balanced" => Ayuda al desbalanceo de clases

Modelado baseline & optimización de balanceo de target

	SMOTE	SMOTETomek	UnderSampling	SMOTE + UnderSampling Train	SMOTE + UnderSampling Test
Accuracy	0.66	0.65	0.67	0.96	0.86
F1 Score (macro)	0.31	0.31	0.52	0.88	0.48
Precision (macro)	0.31	0.31	0.46	0.86	0.53
Recall (macro)	0.36	0.36	0.86	0.9	0.45
Matriz de Confusión	[[88764 16724 6455 11270] [2910 1296 714 1745] [2578 1134 905 1752] [2209 883 825 2520]]	[[88325 16604 6930 11354] [2910 1226 787 1742] [2567 1109 949 1744] [2193 880 836 2528]]	[[77414 18817 15751 11231] [123 6165 170 207] [56 125 5976 212] [41 127 186 6083]]	[[120232 998 1037 946] [640 5746 130 149] [514 107 5542 206] [373 114 155 5795]]	[[29410 462 452 479] [1050 400 89 128] [954 89 394 155] [793 90 133 593]]

- SMOTE o SMOTETomek solos dan resultados muy pobres, evidenciando que solo generar sintéticos sin reducir la clase mayoritaria no es suficiente.
- El UnderSampling solo mejora el recall macro (0.86) respecto a SMOTE, pero sigue siendo pobre en F1 y Precisión.
- La combinación SMOTE + UnderSampling en entrenamiento logra unas metricas bastante aceptables, pero en test baja a 0.86, indicando overfitting => lo mejoraremos en la optimización del modelo.

Modelado baseline & optimización de balanceo de target

Comparativa de parametros para el balanceo SMOTE + UnderSampling en test para multiclase y binario

Reporte de clasificación:

	precision	recall	f1-score	support
0	0.99	0.98	0.98	123213
1	0.82	0.86	0.84	6665
2	0.81	0.87	0.84	6369
3	0.82	0.90	0.86	6437
accuracy			0.96	142684
macro avg	0.86	0.90	0.88	142684
weighted avg	0.96	0.96	0.96	142684

Reporte de clasificación:

	precision	recall	f1-score	support
0		0.92	0.95	0.94
1		0.62	0.47	0.54
accuracy				0.89
macro avg	0.77	0.71	0.74	35671
weighted avg	0.88	0.89	0.88	35671

Optimización de modelo & hiperparámetros

Baseline (RF / SMOTE+UnderSampling)



1ra optimización

model	best_score	best_params
LogisticRegression	0.679558	{'classifier': LogisticRegression(class
RandomForestClassifier	0.781272	{'classifier': RandomForestClassifier(
XGBClassifier	0.758375	{'classifier': XGBClassifier(class_w
GradientBoostingClassifier	0.755361	{'classifier': GradientBoostingClassifi
DecisionTreeClassifier	0.723503	{'classifier': DecisionTreeClassifier(ra
AdaBoostClassifier	0.733538	{'classifier': AdaBoostClassifier(n

Modelos: logistic Regression, random_forest, xgb, gbct, tree, ada_boost

Pipeline: PCA + Scaler + Select KBest + classifier

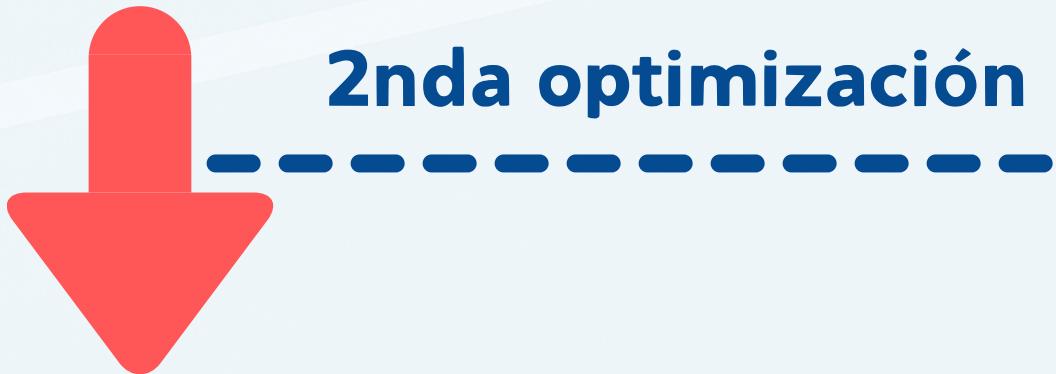
CV: GridSearchCV (CV=5)

Hiperparametros: 219 combinaciones x 5 splits = 1095 combinaciones



Optimización de modelo & hiperparámetros

RF (Parámetros de 1ra optimización)



Random Forest Classifier =>

Score 0.785

n_estimators: 300

max_features: 'log2'

max_depth: None

Modelos: Random Forest
(Class_weight =balanced)

Pipeline: classifier (como es RF, no necesita nada más)

cv: RandomizedSearchCV (cv=5, n_iters = 50)

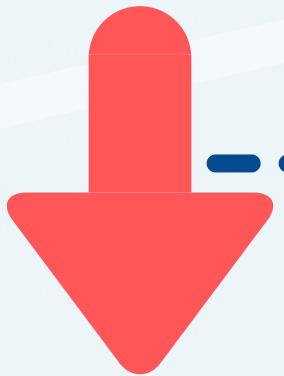
Hiperparametros: $24 \times 5 = 120$

- max_depth: [3,4, 5, None]
- n_estimators: [100,200,300]
- max_features:['sqrt','log2']



Optimización de modelo & hiperparámetros

RF (Parámetros de 2da optimización)



3ra optimización

Random Forest Classifier =>

Score 0.785

n_estimators: 350

max_features: 'log2'

max_depth: None

No somos capaces de optimizar
mas nuestro modelo

Modelos: Random Forest
(Class_weight =balanced)

Pipeline: classifier (como es RF, no necesita nada más)

CV: GridSearchCV
(cv=5)

Hiperparametros: $16 \times 5 = 80$

- max_depth: [None]
- n_estimators: [300, 350]
- max_features:['log2']
- Criterion: ['gini', 'entropy']
- min_samples_split: [2,5]
- min_samples_leaf: [1,2]



Optimización de modelo & hiperparámetros



RF (Parámetros de 3ra optimización)



Aplicamos a nuestro train

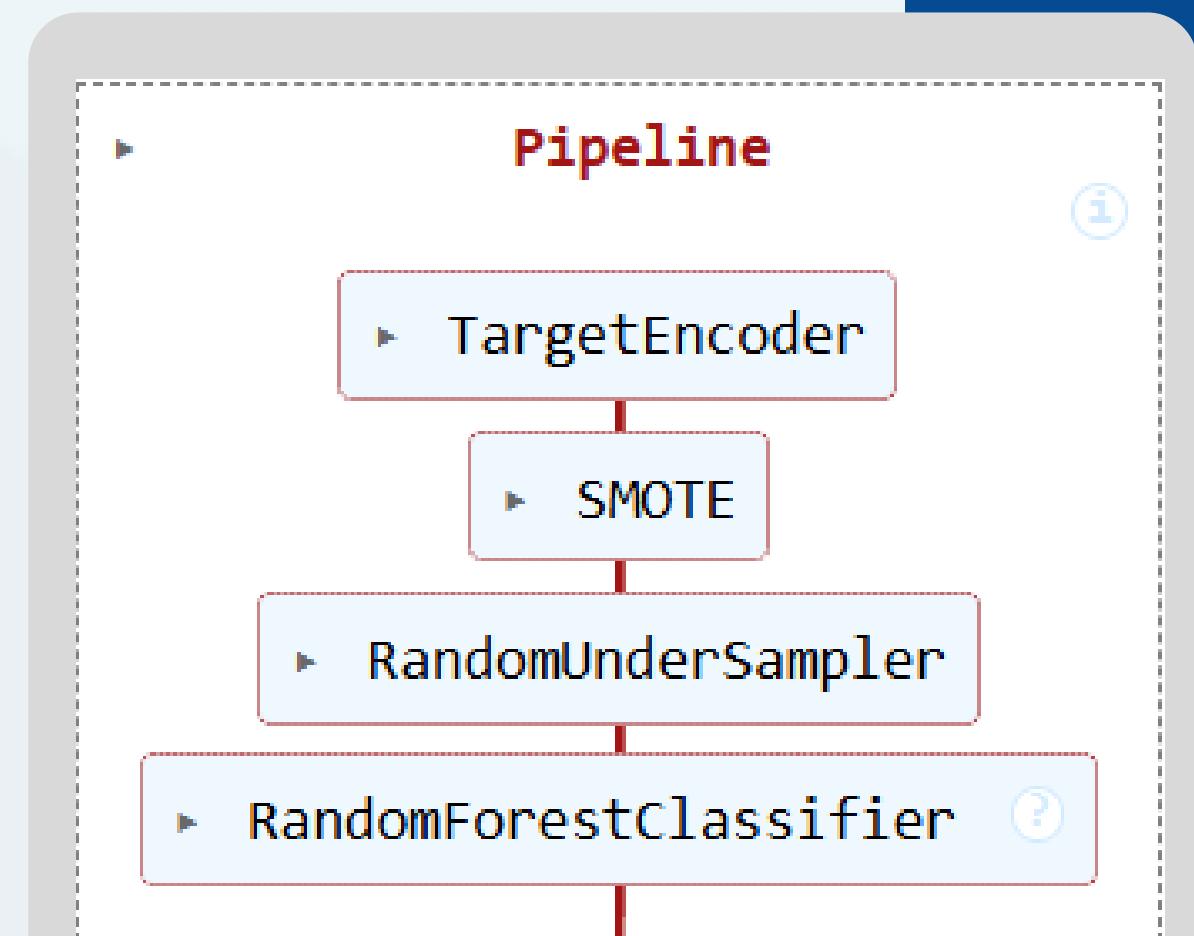
RESULTADOS TRAIN

Matriz de Confusión:

```
[[120322  968  995  928]
 [ 661  5734  123  147]
 [ 537   114  5510  208]
 [ 380   111  158 5788]]
```

		Reporte de clasificación:			
		precision	recall	f1-score	support
	0	0.99	0.98	0.98	123213
	1	0.83	0.86	0.84	6665
	2	0.81	0.87	0.84	6369
	3	0.82	0.90	0.86	6437
	accuracy			0.96	142684
	macro avg	0.86	0.90	0.88	142684
	weighted avg	0.96	0.96	0.96	142684

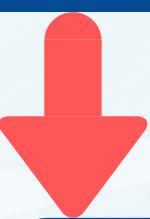
- El modelo clasifica extremadamente bien la clase mayoritaria (0) con un recall de 98% y un accuracy del 96%.
- Las clases minoritarias (1, 2 y 3) ahora se reconocen muy bien, los métodos de balanceo han tenido un efecto positivo
- Las métricas macro (≈ 0.88) indican que el modelo no solo se apoya en la clase 'No medal'. El buen F1 macro demuestra que el modelo distingue bien entre todas las clases, no solo la dominante.



- max_depth: None
- n_estimators: 350
- max_features:'log2'
- Criterion: 'gini'
- min_samples_split: 2
- min_samples_leaf: 1

Optimización de modelo & hiperparámetros

RF (Parámetros de 3ra optimización)



Aplicamos a nuestro test

RESULTADOS TEST

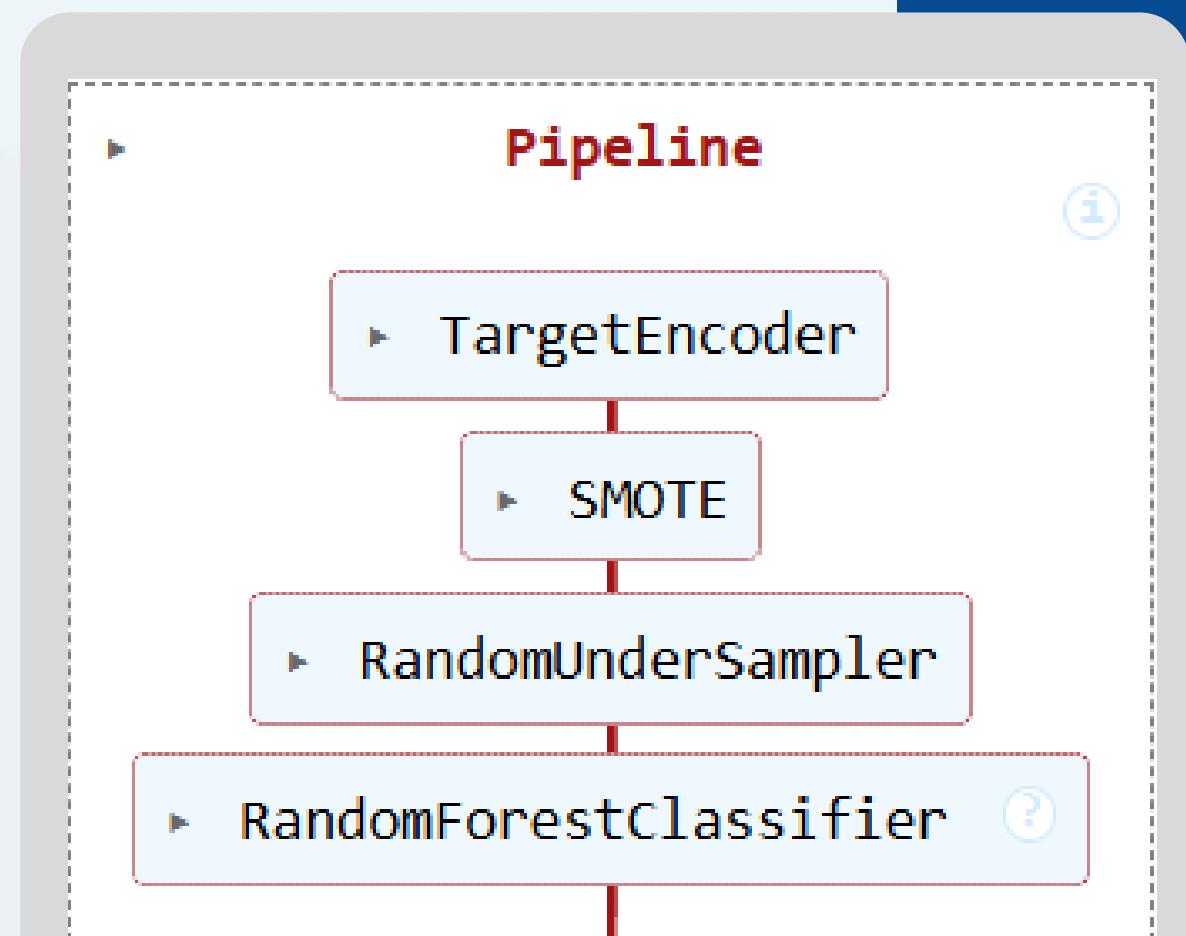
Matriz de Confusión:

```
[[29456  455  467  425]
 [ 1045  418   86 118]
 [  947   85  425 135]
 [  765   92  134 618]]
```

Reporte de clasificación:

	precision	recall	f1-score	support
0	0.91	0.96	0.93	30803
1	0.40	0.25	0.31	1667
2	0.38	0.27	0.31	1592
3	0.48	0.38	0.43	1609
accuracy			0.87	35671
macro avg	0.54	0.46	0.50	35671
weighted avg	0.85	0.87	0.85	35671

- El modelo sigue funcionando muy bien para la clase mayoritaria (0), pero falla fuerte en las minoritarias. La clase 0 mantiene un recall de 0.96, pero las clases 1, 2 y 3 bajan muchísimo (entre 0.25 y 0.38). Esto indica que el pipeline entrenado no generaliza bien el rebalanceo SMOTE a datos reales.
- La brecha entre train (F1 macro ≈ 0.88) y test (F1 macro ≈ 0.50) confirma sobreajuste.
- Se hace muy difícil predecir con precisión entre diferentes medallas, aunque es mucho más fácil predecir si un atleta la ganara o no.



Hiperparámetros:

- max_depth: None
- n_estimators: 350
- max_features:'log2'
- Criterion: 'gini'
- min_samples_split: 2
- min_samples_leaf: 1

Optimización de modelo & hiperparámetros



Modelo final binario

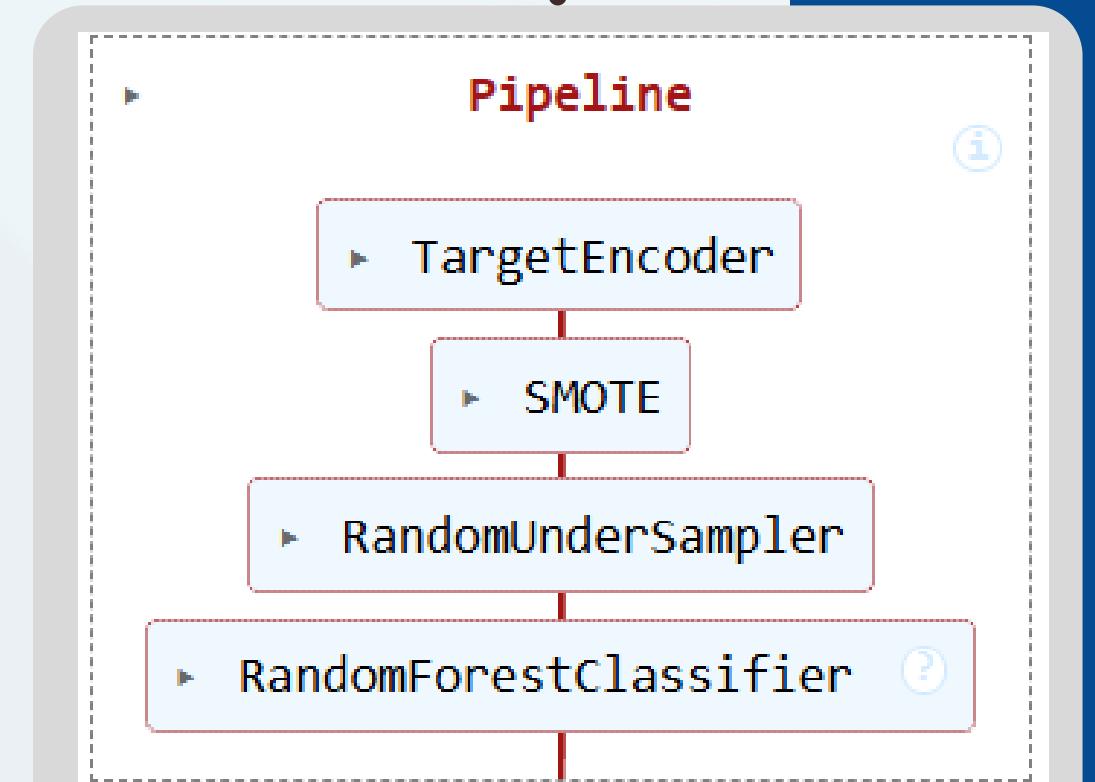
Clasificador Binario

RESULTADOS TEST y comparativa con multiclase

Matriz de Confusión:
[[29333 1470]
[2574 2294]]

Reporte de clasificación:					
	precision	recall	f1-score	support	
0	0.92	0.95	0.94	30803	
1	0.61	0.47	0.53	4868	
accuracy			0.89	35671	
macro avg	0.76	0.71	0.73	35671	
weighted avg	0.88	0.89	0.88	35671	

- Accuracy **0.89** y F1 **0.88** indican que el modelo predice correctamente la mayoría de los casos y mantiene un buen equilibrio entre precision y recall.
- La clase minoritaria (medalla) tiene un recall de **0.47**, lo que significa que todavía se pierden bastantes medallas, aunque la precisión de **0.61** indica que cuando predice medalla, lo hace bien la mayoría de las veces.
- La matriz de confusión refleja que el modelo sigue confundiendo algunas medallas con no medallas (**2574 falsos negativos**), pero comparado con la multiclasicación, la simplificación a binaria mejora claramente las métricas globales, ya que no se penaliza por confundir entre clases de medalla.

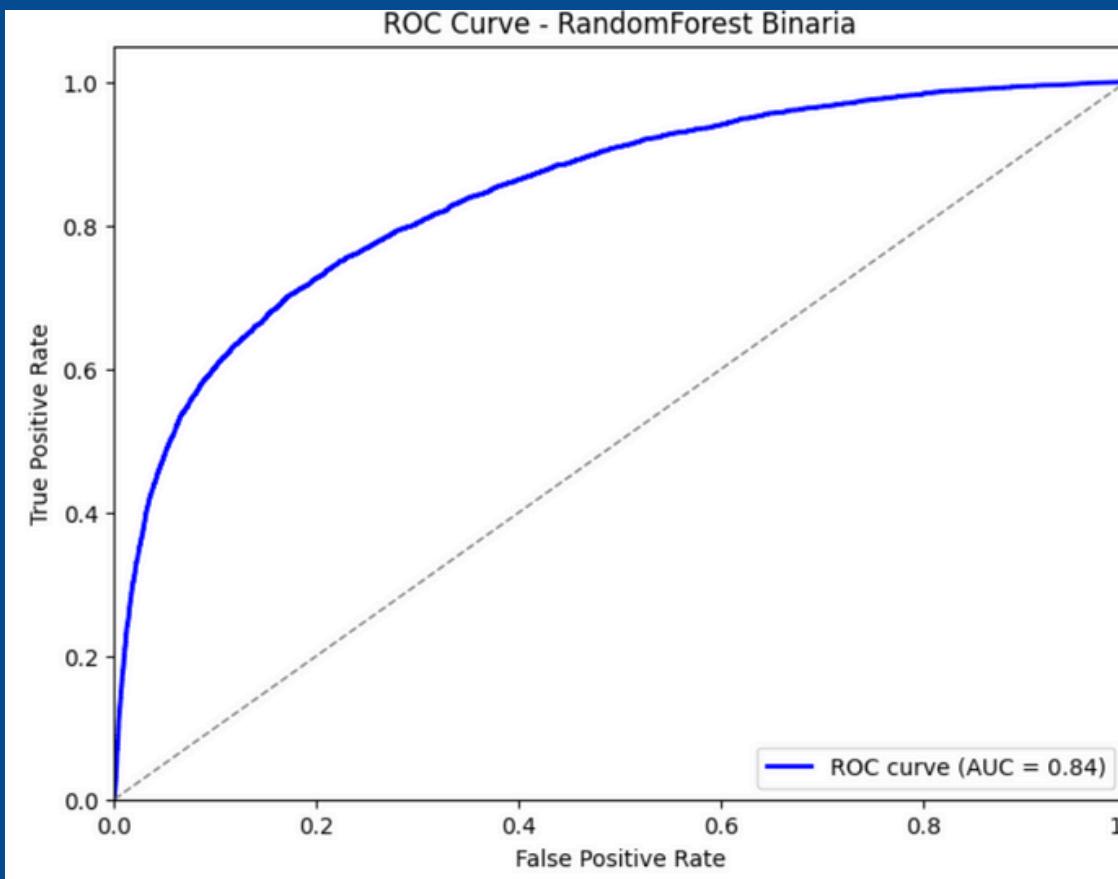


Hiperparámetros:

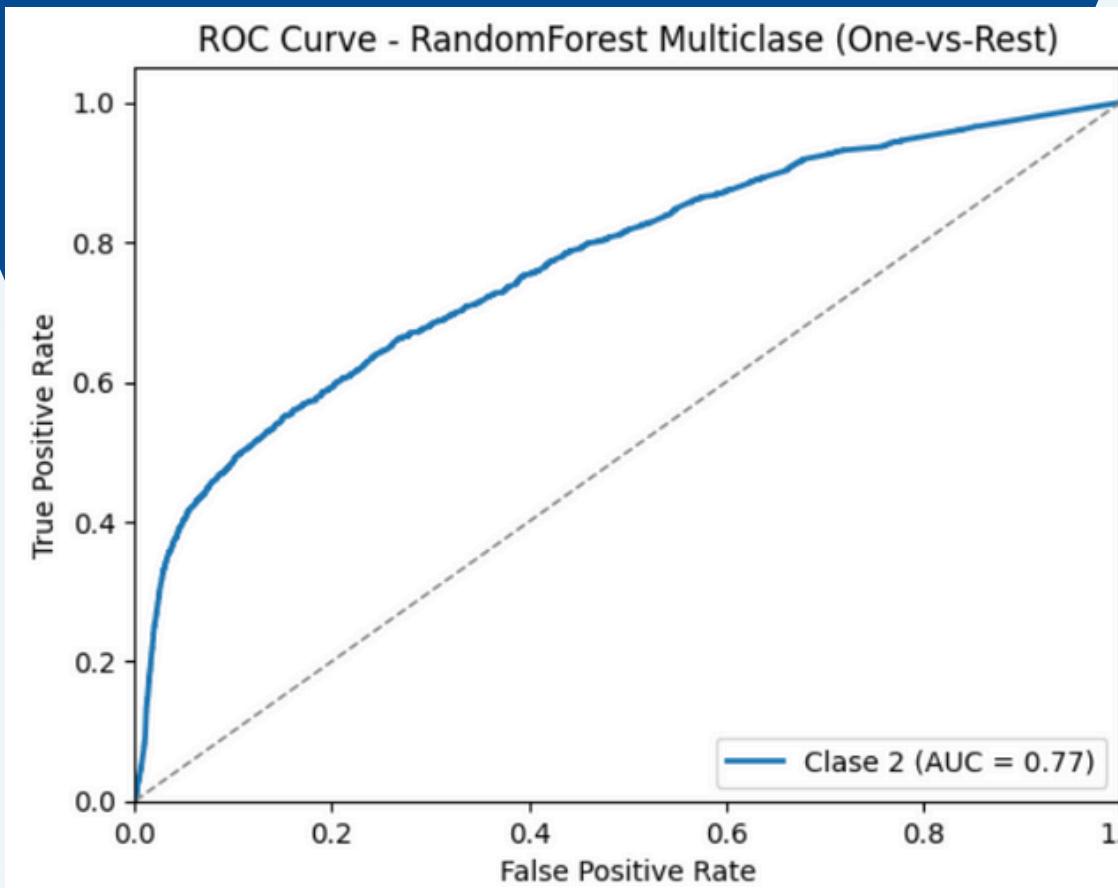
- max_depth: None
- n_estimators: 350
- max_features:'log2'
- Criterion: 'gini'
- min_samples_split: 5
- min_samples_leaf: 1

Curva ROC binario & multiclase

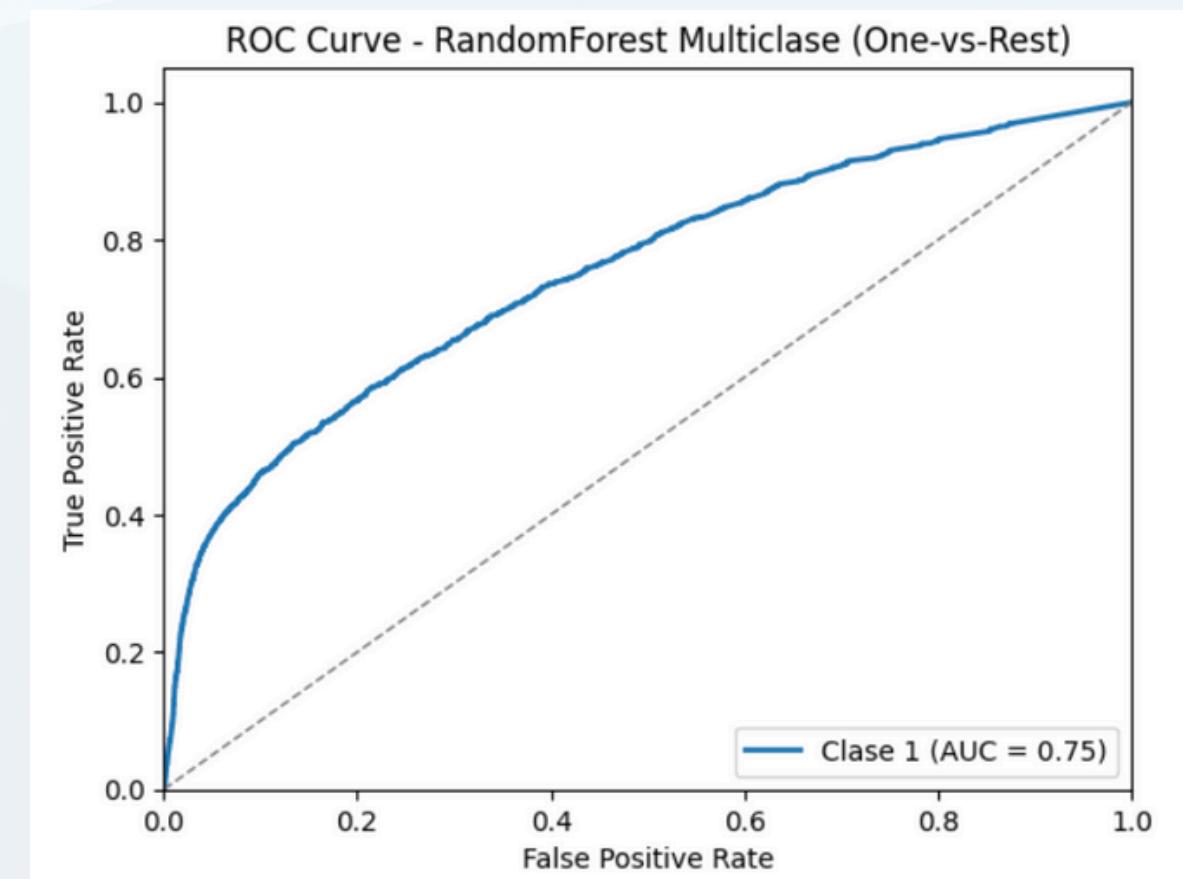
binaria



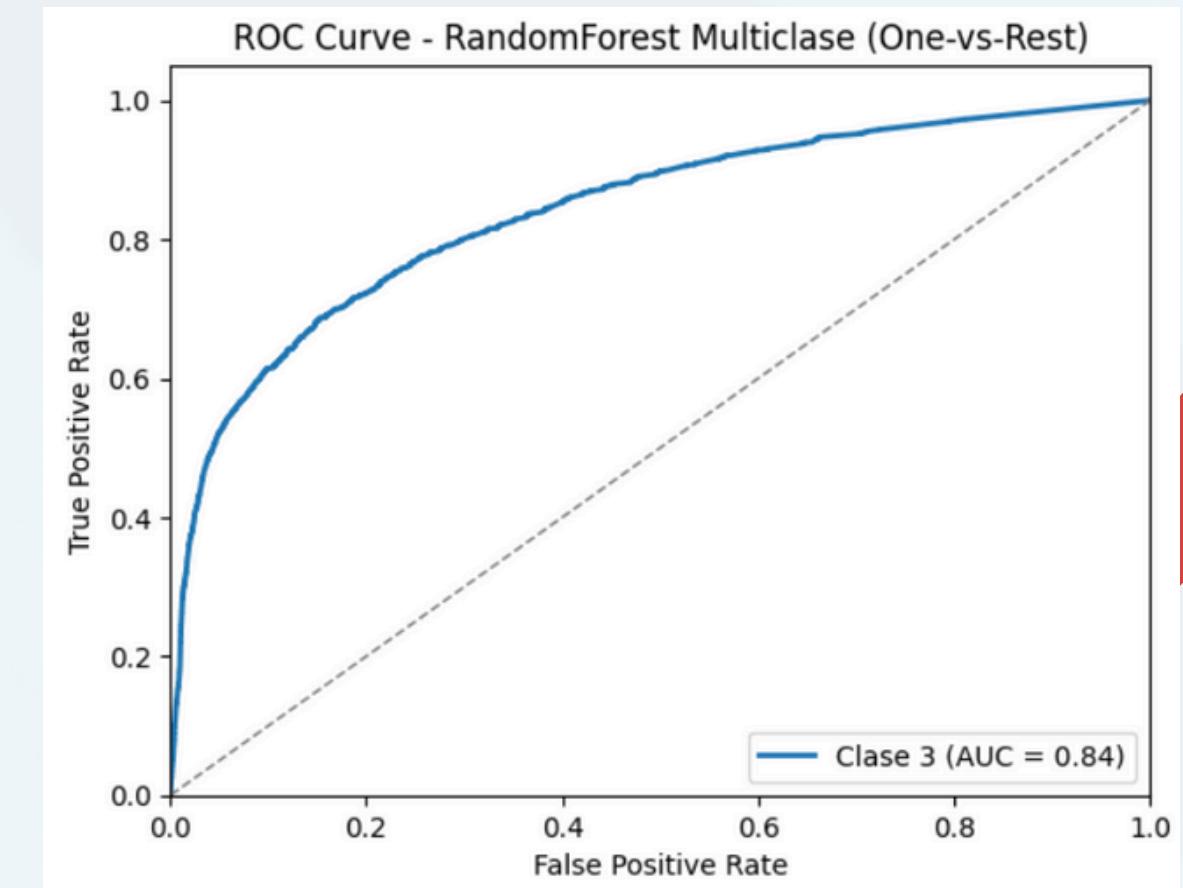
Clase 2 (plata)



Clase 1 (oro)



Clase 3 (bronce)



¡Gracias!



Y si me das una semana más?....

Optimizar balanceo

Optimizar aún mas hiperparametros utilizando
Optuna

buscar otros datos que aporten correlación
con las medallas (horas de entrenamiento,
inversión de la federación).

Trabajar más las features important.

Y ahora el momento de la verdad...



Streamlit

THE
BRIDGE

