Análisis y resolución de preguntas:

Reporte de clasificación:

Accuracy del mejor modelo: 0.8604651162790697

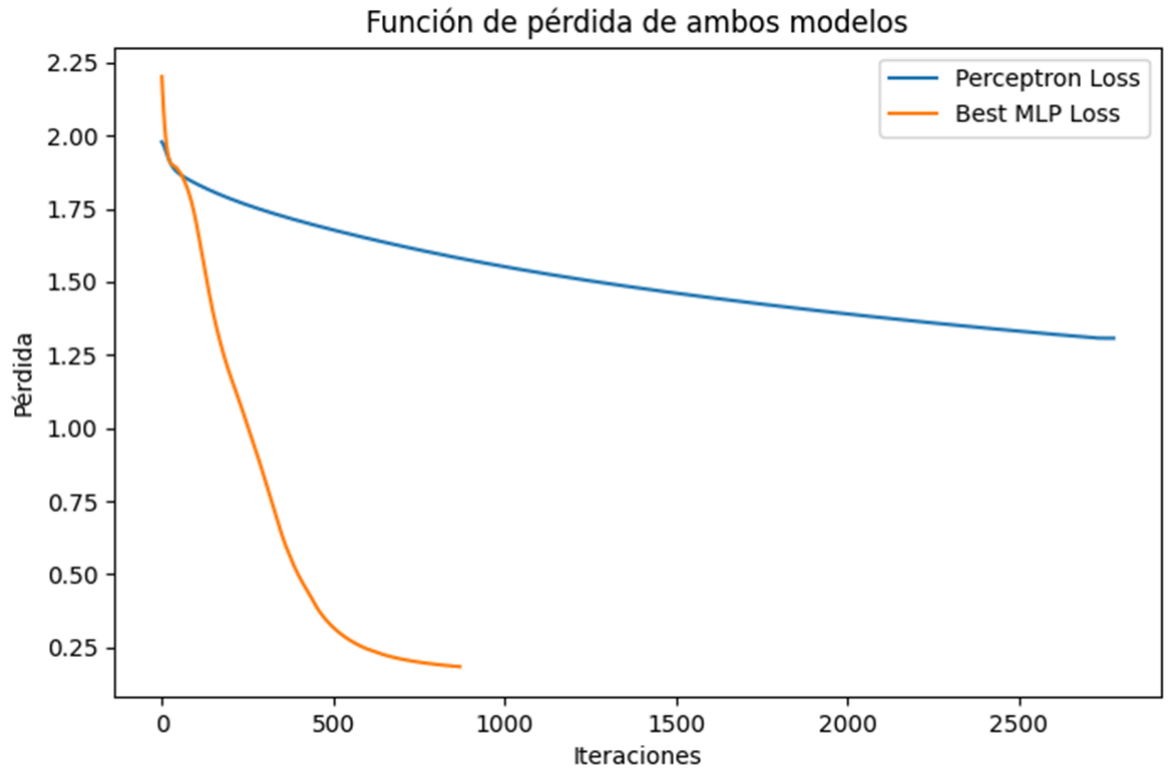
	precision	recall	f1-score	support
0	0.82	0.93	0.88	15
1	0.91	0.71	0.80	14
2	0.87	0.93	0.90	14
accuracy			0.86	43
macro avg	0.87	0.86	0.86	43
weighted avg	0.87	0.86	0.86	43

La optimización permitió mejorar significativamente el desempeño del modelo, pasando de una accuracy de 58% a 86%.

- Se observa una mejora en las métricas de precisión, recall y f1-score para todas las clases. Por ejemplo, la clase 1 pasó a tener un f1-score de 0.80 en el modelo optimizado.
- El uso de una arquitectura más profunda (tres capas ocultas de 50 neuronas cada una) junto con el optimizador adam y una tasa de aprendizaje constante de 0.001 resultó ser la configuración óptima para este problema.
- La búsqueda de hiperparámetros demostró ser crucial para captar patrones complejos en los datos, mejorando la capacidad de generalización del modelo.

En resumen, la optimización a través de GridSearchCV logró transformar un modelo inicial con desempeño modesto en uno que clasifica significativamente mejor los niveles de PIB. Esto respalda la importancia de ajustar los hiperparámetros y validar cuidadosamente cada configuración en problemas de clasificación supervisada. ¿Hay algo más que desees discutir o ajustar en esta sección?

Interpretación:



En la curva de color azul se aprecia la evolución de la pérdida del modelo Perceptron base, mientras que en la curva naranja se observa la del Best MLP, resultado de la búsqueda de hiperparámetros. Al inicio, ambos modelos parten de un valor de pérdida superior a 2.0, lo cual es común en configuraciones iniciales cuando los pesos del modelo aún no han sido ajustados. Sin embargo, el Best MLP (naranja) desciende de forma mucho más pronunciada en las primeras iteraciones, mostrando un aprendizaje rápido y eficiente.

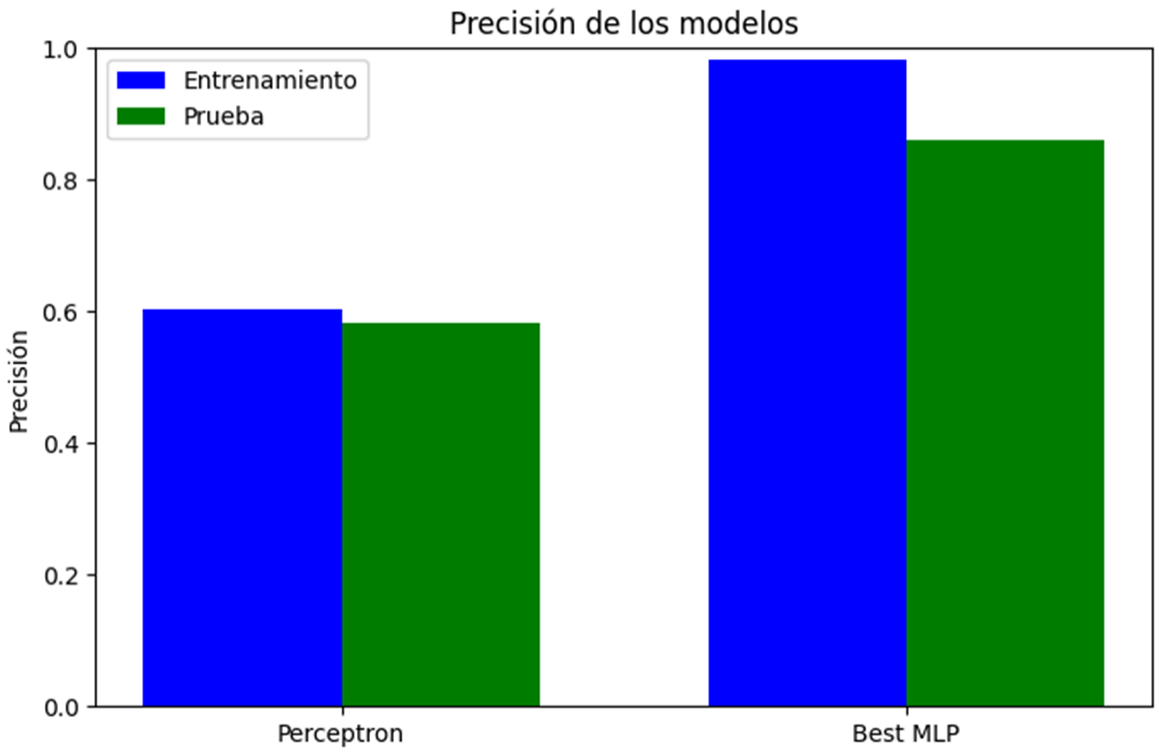
A medida que transcurren las iteraciones, la curva del Perceptron (azul) desciende más lentamente, estabilizándose alrededor de 1.2 o 1.3. En contraste, la curva del Best MLP llega a valores cercanos a 0.2, lo que evidencia un ajuste más profundo de los datos de entrenamiento. Este comportamiento sugiere que la configuración del Best MLP —con un optimizador más avanzado (adam), un número mayor de capas y neuronas, y una tasa de aprendizaje adecuada— permite al modelo converger de manera más efectiva, reduciendo la pérdida de forma sustancial.

En términos de velocidad de convergencia, el Perceptron (con sgd y learning_rate='adaptive') no logra alcanzar niveles de pérdida tan bajos, indicando que su capacidad de representación y la efectividad de su optimizador son menores en comparación con el Best MLP. Al finalizar las iteraciones, la curva del Perceptron sigue un ritmo más lento de descenso y se mantiene lejos del mínimo alcanzado por el Best MLP.

Finalmente, estos resultados de la función de pérdida se correlacionan con la precisión alcanzada por cada modelo en el conjunto de prueba. Mientras el Perceptron logró alrededor de un 58% de exactitud, el Best MLP alcanzó un 86%, reflejando la importancia de una arquitectura más profunda y un mejor ajuste de los hiperparámetros para mejorar el rendimiento del modelo en la tarea de clasificación.

En esta gráfica se muestran líneas prácticamente planas para ambos modelos, lo que indica que la tasa de aprendizaje se mantiene constante en el valor inicial a lo largo de las iteraciones. A pesar de que en la configuración de los modelos se especificó learning_rate='adaptive' en el Perceptron y learning_rate='constant' en el Best MLP, la forma en que scikit-learn maneja y reporta el atributo learning_rate_init hace que en la gráfica solo se refleje la tasa de aprendizaje inicial (por ejemplo, 0.001), sin capturar los eventuales ajustes internos que podrían realizarse durante el entrenamiento.

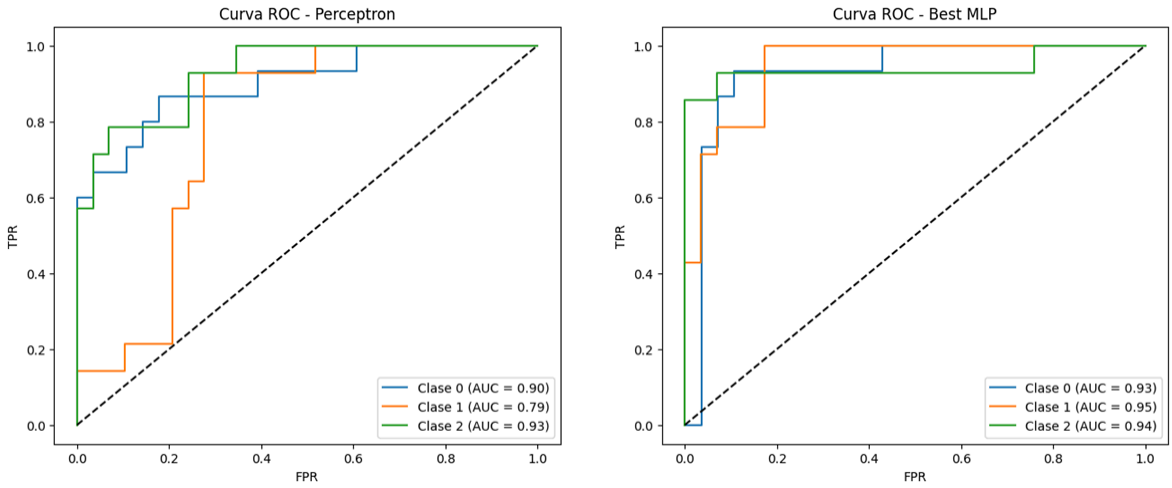
Por ello, las dos líneas se ven superpuestas o muy cercanas, en torno a 0.001, sin cambios visibles. Esto no implica necesariamente que el modelo no esté ajustando su tasa de aprendizaje internamente (especialmente en el caso del Perceptron con learning_rate='adaptive' o del Best MLP con solver='adam'), sino que la función de graficado no accede a la evolución interna de la tasa de aprendizaje. En consecuencia, la curva permanece horizontal y no muestra una variación real a lo largo de las iteraciones.



En la gráfica se presentan dos grupos de barras correspondientes a los dos modelos evaluados: el Perceptron (izquierda) y el Best MLP (derecha). Cada grupo contiene dos barras que representan la precisión en entrenamiento (azul) y en prueba (verde). Esta comparación visual permite apreciar el comportamiento de ambos modelos en términos de su capacidad de aprendizaje y generalización.

Para el Perceptron, se observa que la precisión en entrenamiento (barra azul) alcanza un nivel relativamente alto, lo cual indica que el modelo logra ajustarse bien a los datos con los que fue entrenado. Sin embargo, la precisión en prueba (barra verde) desciende de forma notable, ubicándose cerca del 58%. Esta brecha significativa entre entrenamiento y prueba sugiere un posible sobreajuste, donde el modelo memoriza o se ajusta en exceso a los patrones específicos del conjunto de entrenamiento y no generaliza tan bien a datos nuevos.

En contraste, el Best MLP muestra una precisión alta en entrenamiento, reflejando una gran capacidad de aprendizaje, y mantiene también una precisión elevada en el conjunto de prueba, cercana al 86%. Esta menor diferencia entre entrenamiento y prueba evidencia que, pese a la mayor complejidad del modelo (más capas ocultas, optimizador adam, etc.), el Best MLP no cae en un sobreajuste severo y logra generalizar adecuadamente. La búsqueda de hiperparámetros y la arquitectura más profunda permiten captar patrones más complejos en los datos sin perder la habilidad de desempeñarse bien con información no vista durante el entrenamiento.

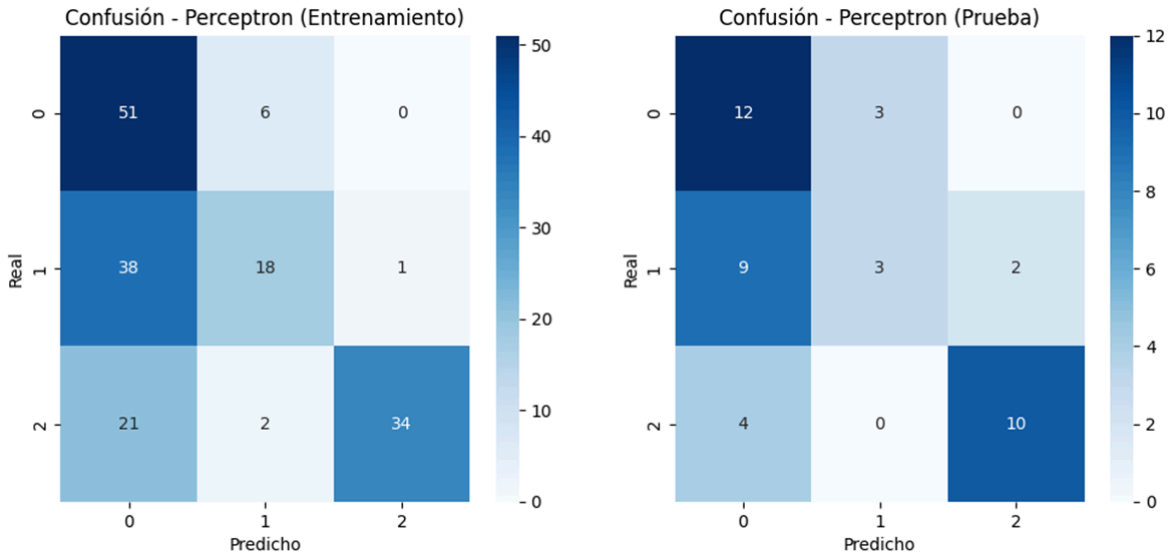


En estas gráficas se comparan las curvas ROC de cada clase (en un esquema One-vs-Rest) para el Perceptron (gráfico izquierdo) y para el Best MLP (gráfico derecho). Cada curva ROC se obtiene al trazar la Tasa de Verdaderos Positivos (TPR) frente a la Tasa de Falsos Positivos (FPR) para distintos umbrales de probabilidad. El área bajo la curva (AUC) cuantifica la capacidad de cada modelo para distinguir correctamente entre las clases.

En el Perceptron (izquierda), se aprecian tres curvas que representan las clases 0, 1 y 2, con valores de AUC de 0.90, 0.79 y 0.93, respectivamente. Esto indica que para la clase 1 el modelo tiene una capacidad de discriminación algo menor (AUC = 0.79), lo cual concuerda con la menor precisión y recall observadas anteriormente. Aun así, las clases 0 y 2 presentan valores de AUC cercanos o superiores a 0.90, lo que muestra un desempeño aceptable para dichas clases.

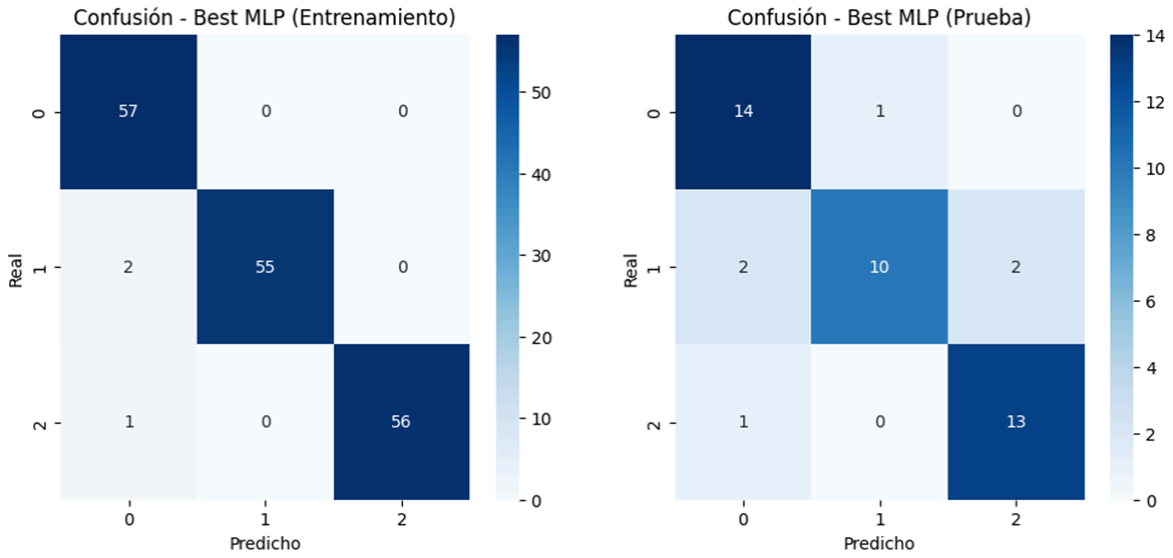
En el Best MLP (derecha), se observan AUC de 0.93, 0.95 y 0.94 para las clases 0, 1 y 2. Estos valores reflejan una capacidad de discriminación más consistente y elevada para cada clase, con todas las AUC por encima de 0.90. Esto corrobora el mayor poder predictivo del Best MLP, que logra separar mejor las instancias de cada clase frente a las demás.

La comparación global evidencia que el Best MLP supera al Perceptron en la mayoría de las clases. El Perceptron muestra un buen comportamiento para clases 0 y 2, pero se queda atrás en la clase 1. El Best MLP, en cambio, mantiene un rendimiento alto y equilibrado para las tres clases, confirmando así la superioridad del modelo optimizado en términos de discriminación y consistencia.



En la matriz de confusión del conjunto de entrenamiento (gráfico izquierdo), se muestran los resultados del Perceptron al clasificar las tres clases (0, 1 y 2). La clase 0 presenta 51 aciertos y 6 errores, lo que indica que el modelo logra reconocerla con relativa solidez. Para la clase 1, se observan 38 instancias correctamente identificadas y 18 clasificadas de forma errónea, evidenciando una dificultad moderada en distinguir esta clase. Finalmente, la clase 2 exhibe 2 errores frente a 34 aciertos, lo cual sugiere que el modelo capta bien sus patrones. Estos resultados implican que el Perceptron, al menos en el entrenamiento, maneja adecuadamente las clases 0 y 2, mientras que la clase 1 se muestra más propensa a confusiones.

En el conjunto de prueba (gráfico derecho), se percibe un descenso en la cantidad de aciertos para todas las clases, un fenómeno habitual al enfrentarse a datos no vistos durante el entrenamiento. La clase 0 logra 12 aciertos y 3 errores, la clase 1 reduce su rendimiento a 8 aciertos y 6 equivocaciones, y la clase 2 mantiene 10 aciertos frente a 4 errores. Estas diferencias subrayan que el Perceptron experimenta cierta merma de desempeño al pasar del entrenamiento a la prueba, un indicio de sobreajuste. Además, la clase 1 sigue siendo la más complicada de predecir, lo que se refleja en una mayor proporción de errores comparada con las clases 0 y 2. En conjunto, estas matrices confirman la capacidad del Perceptron para aprender patrones básicos, pero también ponen de manifiesto sus limitaciones al generalizar, sobre todo en la clase 1.



En la matriz de confusión correspondiente al conjunto de entrenamiento (gráfico izquierdo), se aprecia que el Best MLP clasifica casi a la perfección todas las instancias. La clase 0 cuenta con 57 aciertos y 0 errores, la clase 1 presenta 55 aciertos frente a solo 2 errores, y la clase 2 alcanza 56 aciertos sin ningún desacierto. Estos resultados indican un ajuste casi completo a los datos de entrenamiento, lo que evidencia la alta capacidad de representación y aprendizaje del modelo con la arquitectura y los hiperparámetros óptimos (varias capas ocultas, optimizador adam, etc.).

En la matriz de confusión del conjunto de prueba (gráfico derecho), se mantiene un desempeño sólido, aunque surgen algunos errores al enfrentarse a datos no vistos. La clase 0 obtiene 14 aciertos y 1 error, la clase 1 consigue 10 aciertos y 3 equivocaciones, mientras que la clase 2 logra 13 aciertos y 1 error. Si bien la precisión disminuye ligeramente en comparación con el entrenamiento (lo que es normal en cualquier modelo que abandone el entorno de entrenamiento), el Best MLP demuestra una capacidad de generalización destacable, con un bajo número de errores totales.

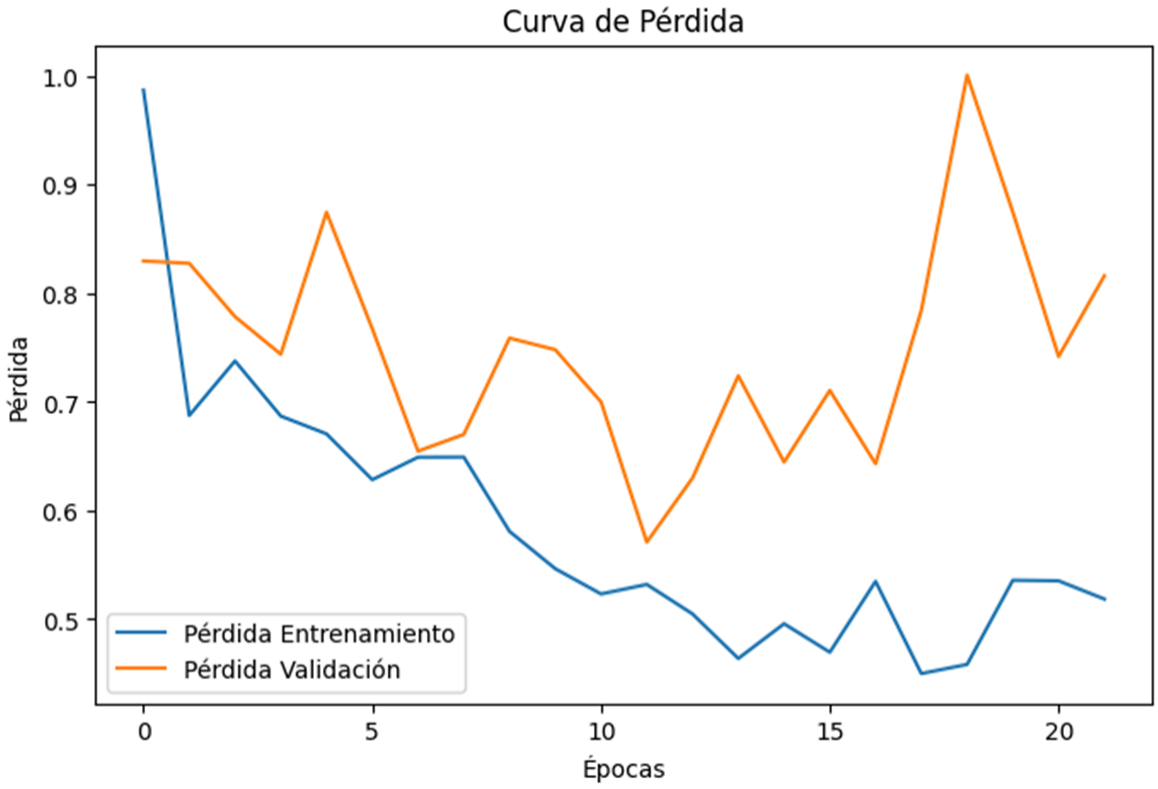
En conjunto, estas dos matrices ilustran la fortaleza del Best MLP para aprender de manera muy efectiva los patrones de los datos de entrenamiento y, a la vez, mantener un nivel de exactitud elevado en el conjunto de prueba. El hecho de que existan algunos errores en la matriz de prueba indica que el modelo no está sobreajustado de forma severa, lo cual se traduce en un equilibrio favorable entre la adaptación a los datos de entrenamiento y la capacidad de predecir instancias nuevas.

Clase	Precision	Recall	F1-Score	Support
0	0.67	0.53	0.59	15
1	0.50	0.71	0.59	14
2	1.00	0.79	0.88	14
Accuracy			0.67	43
Macro avg	0.72	0.68	0.69	43
Weighted avg	0.72	0.67	0.68	43

El modelo de red neuronal profunda se construyó mediante TensorFlow/Keras y se ajustó con Keras Tuner para explorar diferentes configuraciones de capas, neuronas, tasas de dropout, regularización L2 y tasa de aprendizaje. Tras ejecutar 10 ensayos en modo de búsqueda aleatoria (RandomSearch), se encontró un mejor conjunto de hiperparámetros que alcanzó aproximadamente 0.80 de val_accuracy durante la fase de validación interna.

Al evaluar este modelo "óptimo" en el conjunto de prueba, se obtuvo una precisión global de 0.67 (67%). El reporte de clasificación indica un buen desempeño en la clase 2 (100% de precisión y 79% de recall), mientras que las clases 0 y 1 mostraron un equilibrio menor entre precisión y recall. Esta brecha entre la val_accuracy (0.80) y la test accuracy (0.67) sugiere que el modelo podría haberse adaptado de forma específica al conjunto de validación y no generaliza tan bien a datos completamente nuevos.

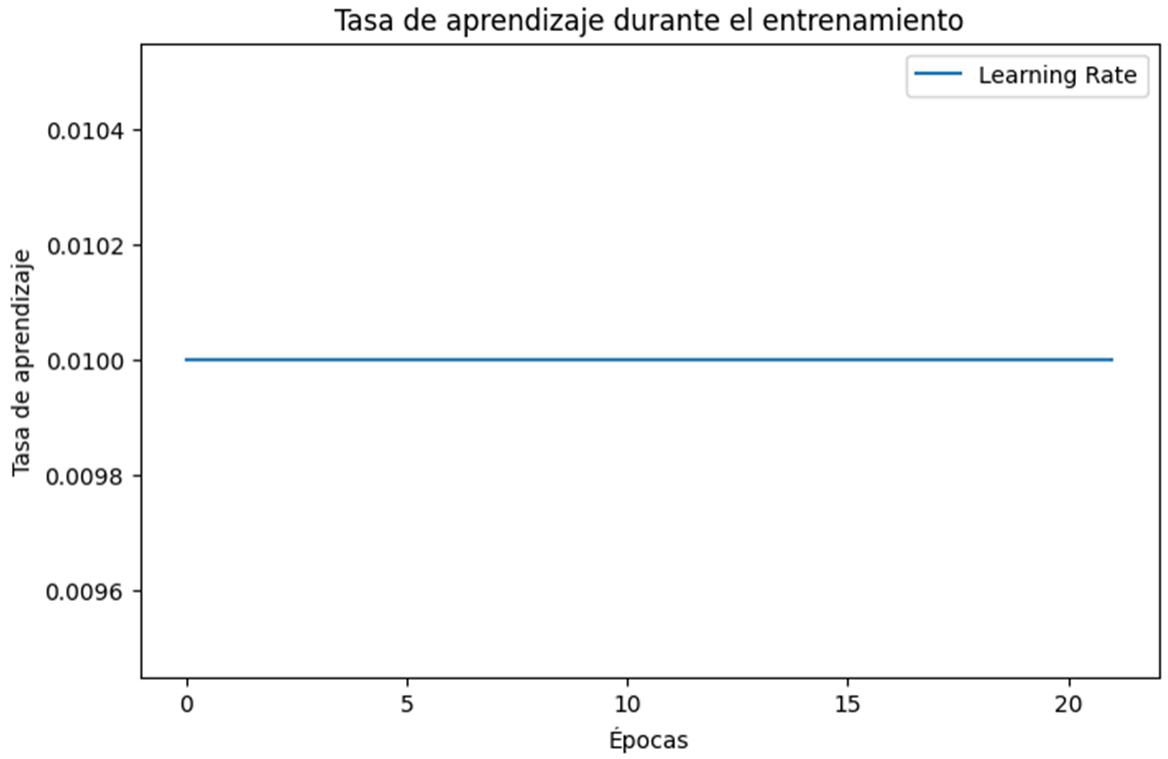
Entre las posibles causas de este desempeño se incluyen el número limitado de ensayos (max_trials=10) en la búsqueda de hiperparámetros y la estrategia de validación interna, que podría no haber sido totalmente representativa. Para mejorar estos resultados, se recomienda ampliar la exploración de hiperparámetros, incrementar los valores de max_trials y/o emplear métodos más exhaustivos (por ejemplo, Bayesian Optimization). Aun así, el modelo demuestra una capacidad de aprendizaje notable, sobre todo en la clase 2, y sirve como base para optimizaciones futuras.



En la curva de color azul se observa cómo la pérdida en entrenamiento desciende de manera constante a lo largo de las épocas, pasando de valores cercanos a 0.9 hacia la región de 0.4. Este descenso continuo indica que el modelo está ajustándose de forma progresiva a los datos de entrenamiento, aprendiendo a predecir cada vez mejor.

Por otro lado, la curva de validación (en naranja) presenta un comportamiento más inestable. Si bien comienza en valores alrededor de 0.8, experimenta fluctuaciones notables y picos que superan 1.0 en algunas épocas, para luego descender otra vez hacia 0.7. Este vaivén sugiere que el modelo no está generalizando de manera consistente en cada época, posiblemente debido a la variabilidad en el conjunto de validación o a ajustes específicos que el modelo hace sobre el set de entrenamiento.

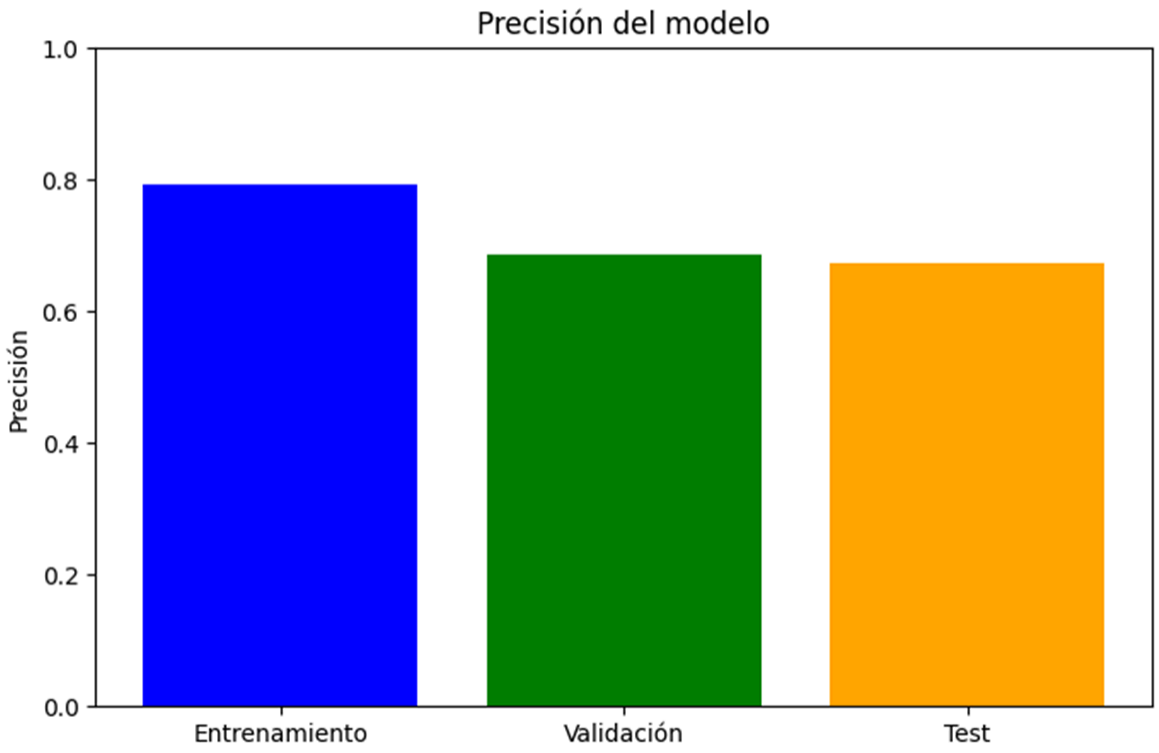
El hecho de que la pérdida de validación termine en un nivel similar o ligeramente superior al inicial, a la vez que la de entrenamiento sigue bajando, apunta a la posibilidad de un sobreajuste (overfitting). En otras palabras, el modelo se adapta progresivamente a las particularidades del entrenamiento, pero no logra sostener ese rendimiento cuando se enfrenta a datos de validación. Esto se confirma al observar la divergencia entre ambas curvas en la segunda mitad del entrenamiento, donde la pérdida de entrenamiento se mantiene relativamente baja, mientras la de validación experimenta altibajos significativos.



En la gráfica se representa la tasa de aprendizaje (learning rate) empleada por el optimizador a lo largo de las épocas de entrenamiento. Se observa una línea plana en torno a 0.01, lo cual indica que, en este caso, la tasa de aprendizaje se mantuvo constante durante todo el proceso.

Este comportamiento se explica porque se estableció un valor fijo (por ejemplo, 0.01) para learning_rate en el optimizador Adam, sin mecanismos de decaimiento o ajuste dinámico. De este modo, el optimizador utiliza el mismo paso de actualización en cada época, sin reducirlo ni incrementarlo en función del avance del entrenamiento.

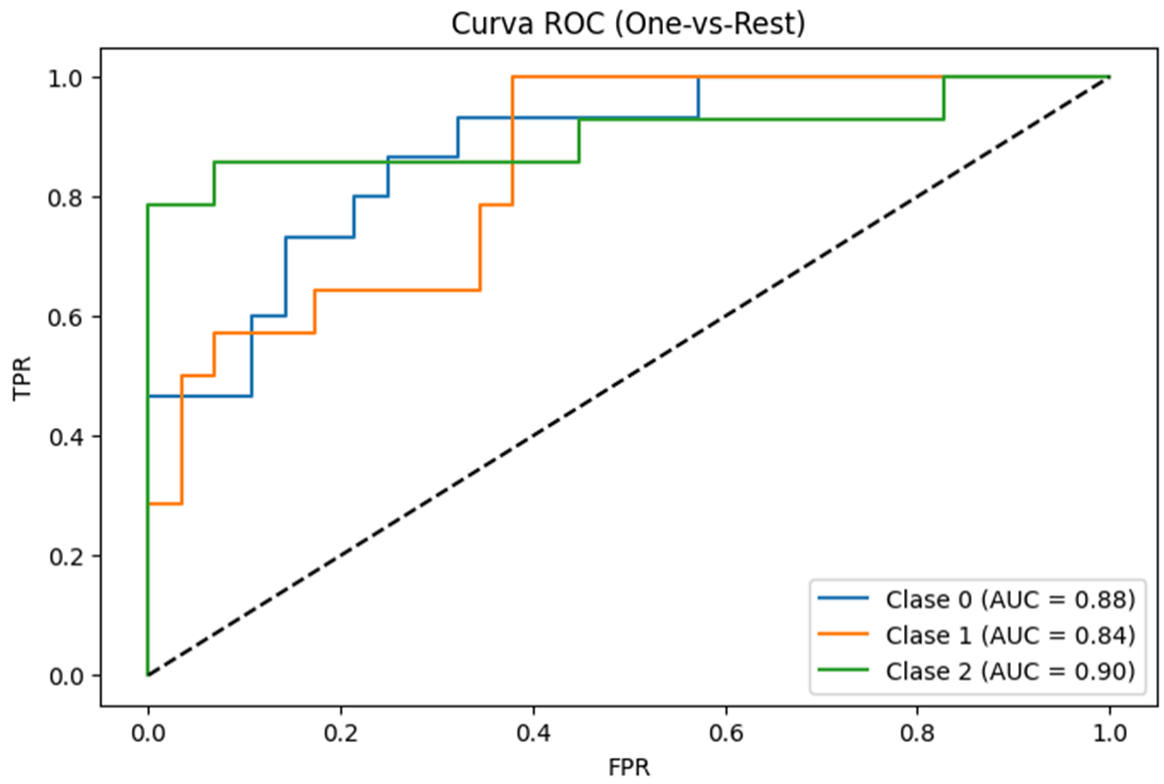
Si bien algunas configuraciones de Adam o de otros optimizadores permiten modificar la tasa de aprendizaje de forma adaptativa, aquí se eligió un esquema sencillo y estático, por lo que la línea permanece invariable y no refleja cambios a lo largo de las iteraciones.



En esta gráfica de barras se muestran tres valores de precisión (accuracy): uno para el conjunto de entrenamiento (azul), otro para la validación interna (verde) y un último para el conjunto de prueba (naranja). Se observa que la barra azul (entrenamiento) es la más alta, lo que indica que el modelo logra un buen ajuste a los datos con los que fue entrenado.

La barra verde, correspondiente a la validación, se sitúa ligeramente por debajo, evidenciando que el desempeño disminuye cuando el modelo se enfrenta a datos no vistos durante la fase de entrenamiento, aunque el descenso no es tan acentuado. Finalmente, la barra naranja (test) muestra un nivel de precisión algo más bajo que la validación, lo que confirma que, al pasar a un conjunto de datos completamente nuevo, el modelo pierde rendimiento adicional.

Este patrón —precisión más alta en entrenamiento, seguida por validación y luego test— es típico de un cierto grado de sobreajuste, donde el modelo se adapta de forma concreta a las particularidades del conjunto de entrenamiento. Aun así, la diferencia entre validación y prueba no es tan grande, lo cual sugiere que el modelo aún mantiene una capacidad razonable de generalización, aunque no alcanza el mismo nivel de desempeño que en entrenamiento.



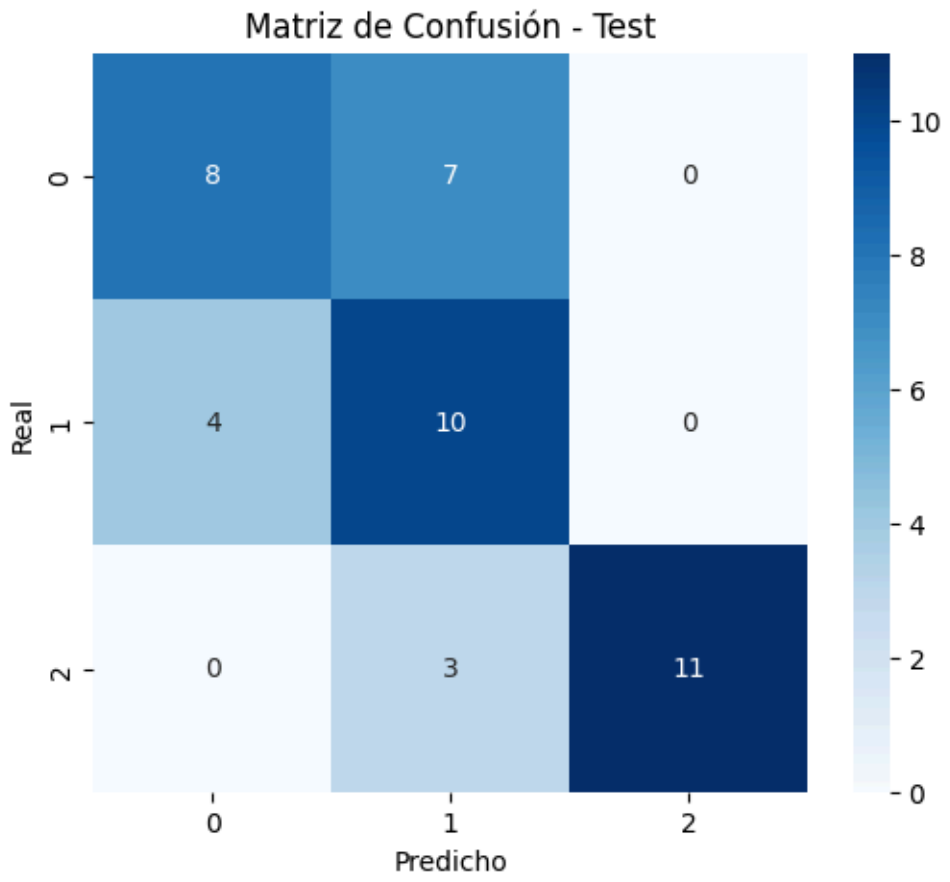
En esta figura se grafican las curvas ROC de cada clase por separado, tratando la clase correspondiente como "positiva" y las demás como "negativas". Cada línea representa la relación entre la Tasa de Verdaderos Positivos (TPR) y la Tasa de Falsos Positivos (FPR) según diferentes umbrales de probabilidad. Se incluyen además los valores de Área Bajo la Curva (AUC) para cada clase:

- Clase 0 (línea azul): Con un AUC de 0.88, el modelo muestra una buena capacidad para discriminar la clase 0 de las restantes.

- Clase 1 (línea naranja): Presenta un AUC de 0.84, lo que indica un rendimiento adecuado, aunque algo menor que las otras clases.

- Clase 2 (línea verde): Es la mejor diferenciada, con un AUC de 0.90, lo que significa una alta capacidad de identificar correctamente la clase 2 frente a las demás.

En general, AUC superiores a 0.80 se consideran buenas, por lo que los valores obtenidos (0.88, 0.84 y 0.90) reflejan un modelo razonablemente sólido en la distinción de cada clase. La ligera variación entre clases señala que el desempeño del modelo no es completamente uniforme; por ejemplo, la clase 1 tiene una discriminación algo menor (AUC=0.84). Aun así, todas las curvas se mantienen alejadas de la diagonal (representada por la línea discontinua) que simboliza un modelo aleatorio, evidenciando una capacidad de clasificación notable.



En esta matriz se analizan las predicciones del modelo en el conjunto de prueba, comparando la clase real (eje vertical) con la clase predicha (eje horizontal) para cada instancia. Observamos tres filas y tres columnas que representan las clases 0, 1 y 2. En la diagonal principal (celdas 0,0; 1,1; 2,2) se muestran los aciertos del modelo, mientras que las demás celdas indican los errores de clasificación.

La primera fila (clase 0 real) presenta 8 aciertos al predecirla como clase 0 y 7 errores cuando fue clasificada como clase 1. Esto significa que, en ocasiones, el modelo confunde la clase 0 con la 1, aunque no la confunda con la clase 2. La segunda fila (clase 1 real) exhibe 10 aciertos, 4 instancias asignadas equivocadamente a la clase 0 y ninguna asignada a la clase 2. Por último, la tercera fila (clase 2 real) cuenta con 11 instancias clasificadas correctamente y 3 ejemplos que el modelo confunde con la clase 1, sin confusiones con la clase 0.

En conjunto, estos números reflejan un desempeño mayoritario de aciertos para cada clase, pero también señalan un patrón de confusión particular: la tendencia a confundir la clase 0 con la 1 y la clase 2 con la 1, mientras que la clase 0 no suele confundirse con la 2, y viceversa. Esta distribución de errores contribuye al resultado global de aproximadamente un 67% de exactitud en el conjunto de prueba, evidenciando que, aunque el modelo distingue razonablemente entre las tres clases, las fronteras entre ellas podrían mejorarse, especialmente entre 0 y 1, y entre 2 y 1.

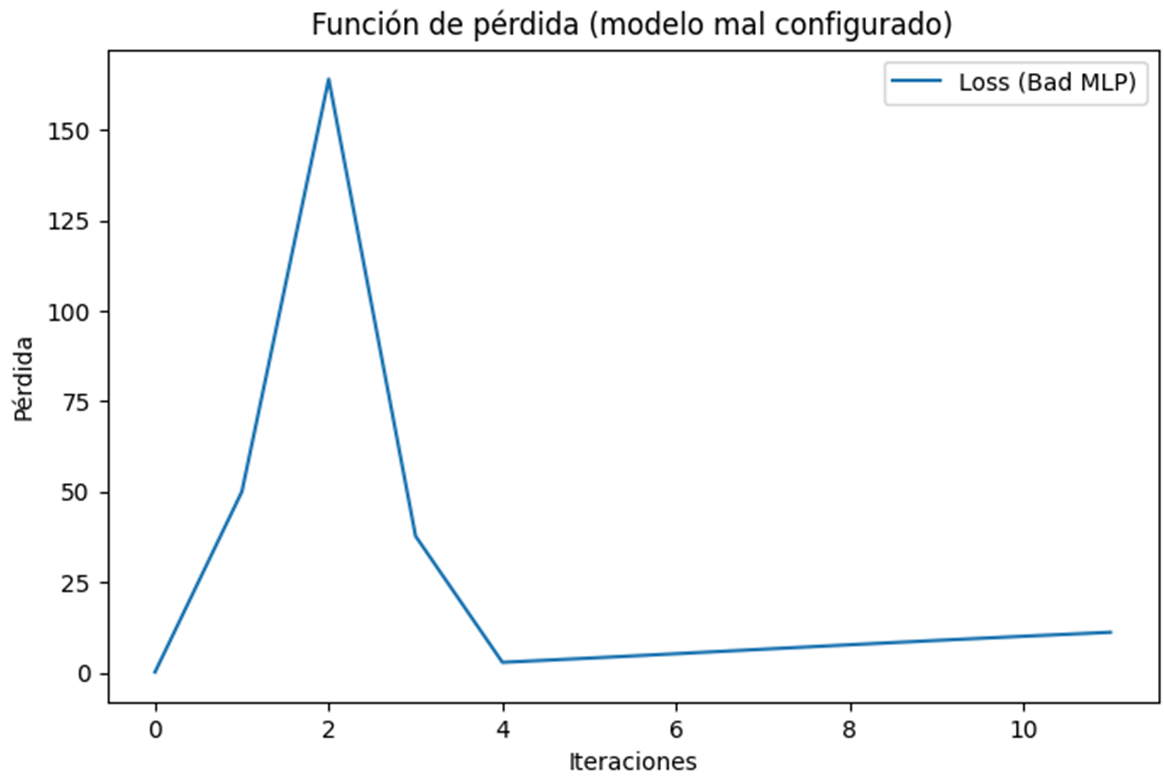
MODELO MAL CONFIGURADO

Accuracy (modelo mal configurado): 0.32558139534883723

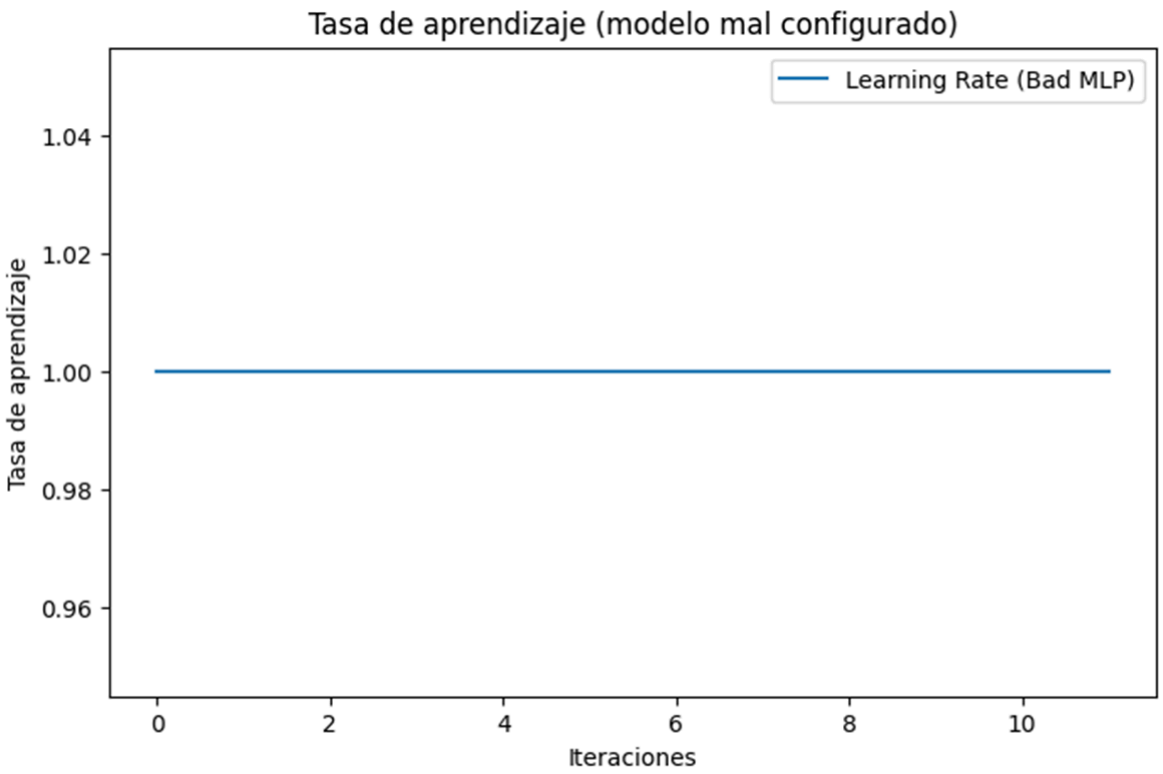
Clase	Precision	Recall	F1-Score	Support
0	0.00	0.00	0.00	15
1	0.00	0.00	0.00	14
2	0.33	1.00	0.49	14
Accuracy			**0.33**	**43**
Macro avg	0.11	0.33	0.16	43
Weighted avg	0.11	0.33	0.16	43

En este fragmento de código se ejemplifica la mala práctica de usar un MLPRegressor para un problema de clasificación, empleando una función de pérdida diseñada para regresión (MSE) en vez de una función propia de clasificación (como cross-entropy). Además, la tasa de aprendizaje se estableció en un valor extremadamente alto (1.0), generando pasos de actualización muy grandes que impiden una convergencia adecuada. Al evaluar este modelo, se observa que clasifica casi todas las instancias como la misma clase (la clase 2), alcanzando así apenas un 32.6% de precisión.

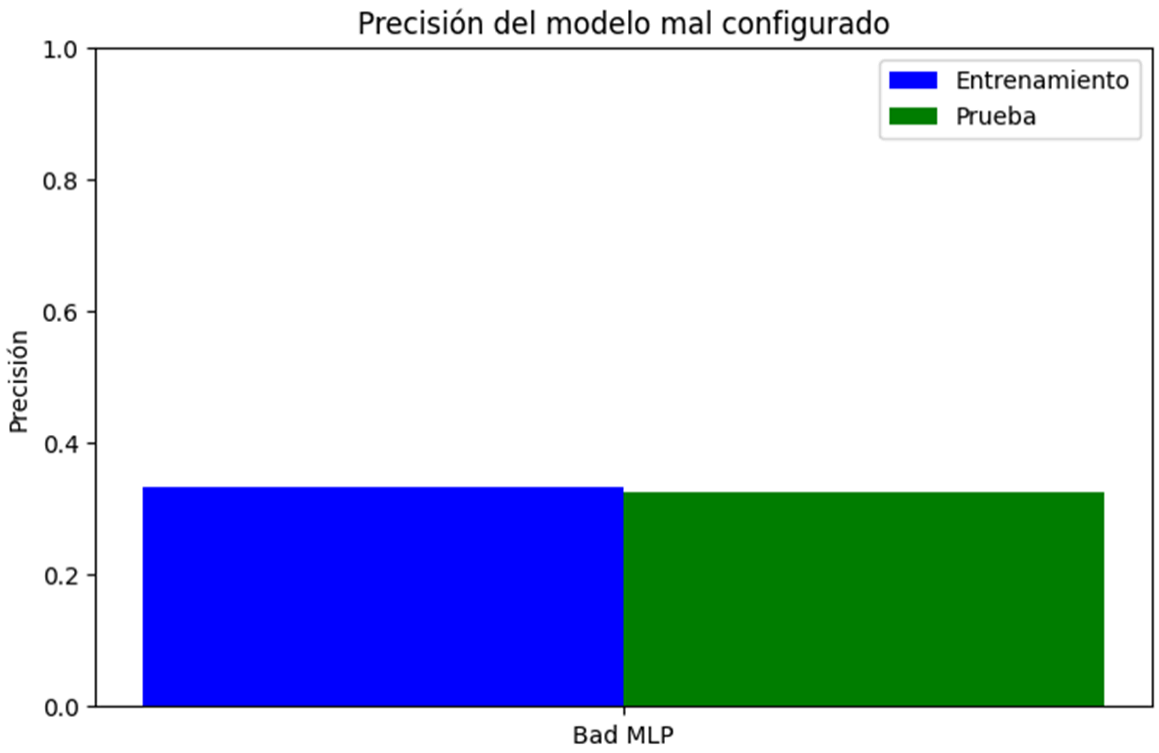
Este resultado deja en evidencia dos graves errores de configuración. Primero, la elección incorrecta del tipo de modelo (regresión en lugar de clasificación) ocasiona que la optimización no se oriente a la separación de clases, sino a la aproximación de valores continuos. Segundo, la tasa de aprendizaje exageradamente alta provoca inestabilidad en la actualización de pesos, ocasionando resultados anómalos como la predicción de una sola clase para la mayoría de las instancias.



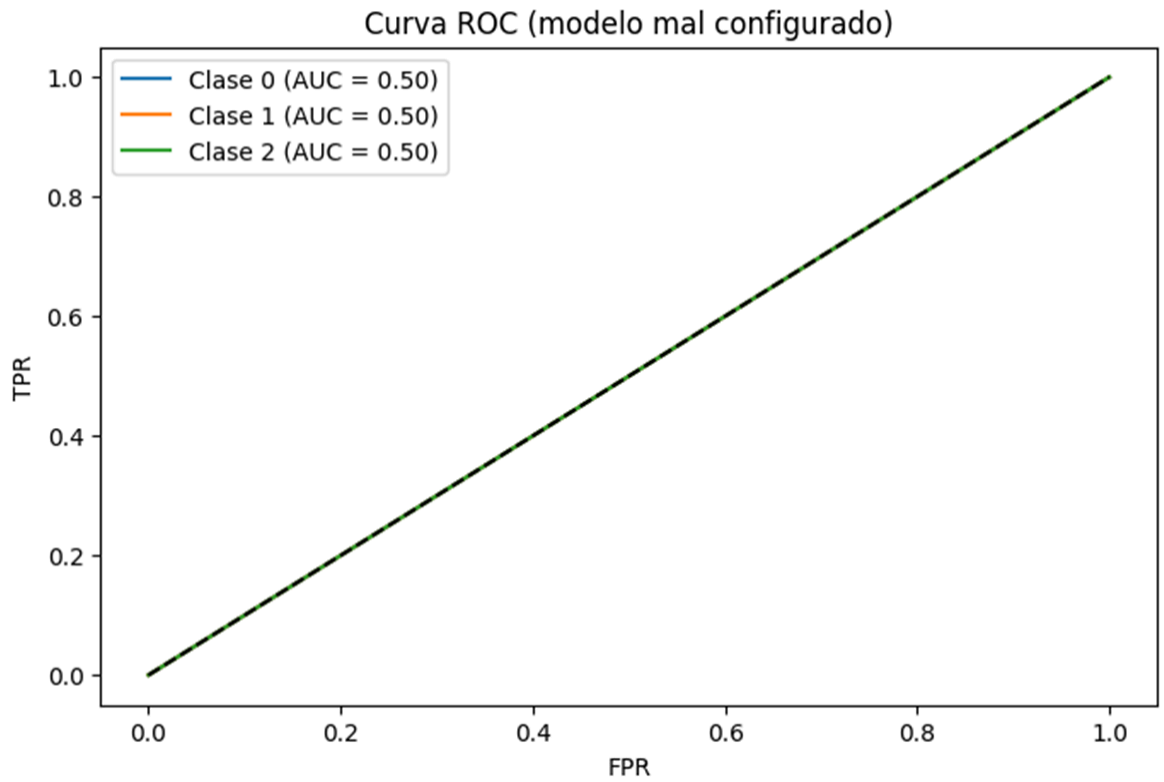
La curva muestra un comportamiento sumamente inestable: inicialmente la pérdida se dispara por encima de 150, luego desciende bruscamente hasta casi cero y, finalmente, vuelve a elevarse de manera moderada. Este patrón indica que los pasos de actualización son excesivamente grandes (tasa de aprendizaje muy alta), provocando saltos radicales en la superficie de optimización. Además, la elección de MLPRegressor para un problema de clasificación intensifica el efecto errático del ajuste, ya que la función de pérdida (MSE) no se alinea con el objetivo de separar clases. En consecuencia, el modelo no converge de forma estable y genera oscilaciones extremas en la pérdida.



Esta gráfica exhibe una tasa de aprendizaje constante en 1.0, un valor sumamente elevado para la mayoría de problemas de redes neuronales. Un learning rate tan alto provoca pasos de actualización enormes en cada iteración, generando comportamientos inestables y dificultando la convergencia. Además, al tratarse de un modelo de regresión usado en clasificación, la elección de esta tasa de aprendizaje extrema empeora el ajuste, resultando en la volatilidad observada en la pérdida y el pobre desempeño final del modelo.

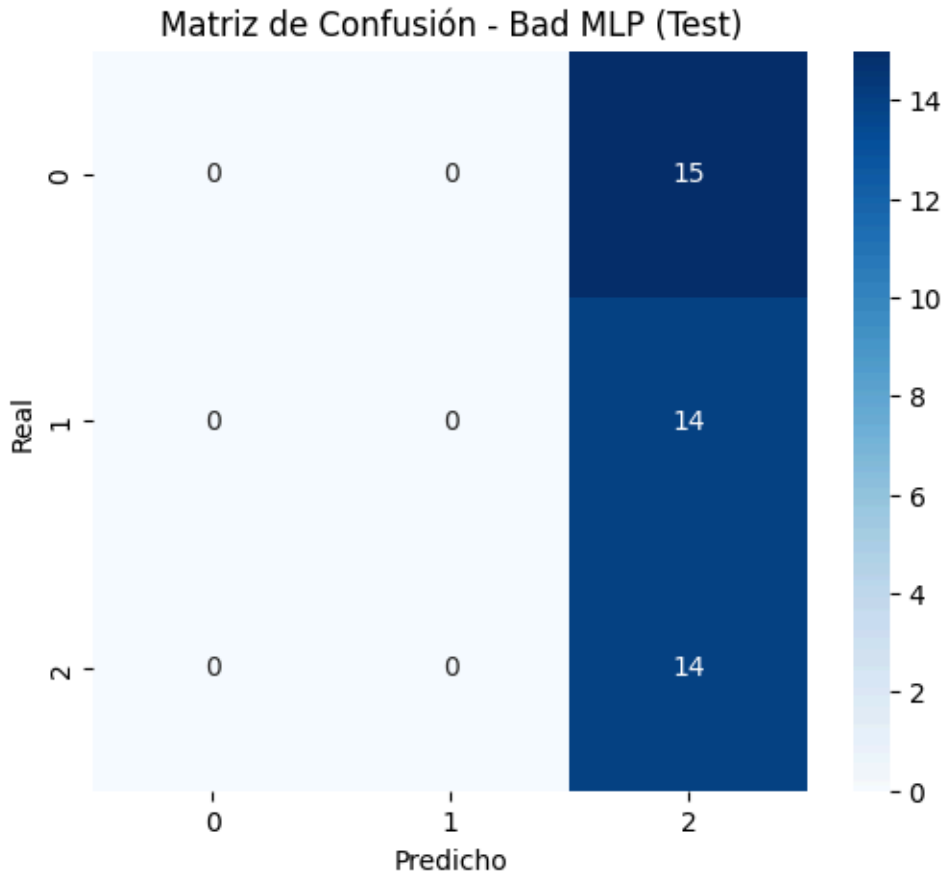


Este gráfico compara la precisión en entrenamiento (barra azul) con la precisión en prueba (barra verde) para el modelo mal configurado. Se aprecia que ambas barras se sitúan en niveles bajos (alrededor de 0.4 en entrenamiento y 0.33 en prueba), lo cual confirma que el modelo no logra separar adecuadamente las clases, ni siquiera en los datos con los que fue entrenado. Esta escasa precisión se explica por el uso de MLPRegressor (orientado a tareas de regresión) en un problema de clasificación y por la tasa de aprendizaje excesivamente alta (1.0), factores que impiden la convergencia y derivan en predicciones pobres en ambas fases (entrenamiento y prueba).



Este gráfico muestra las curvas ROC para cada clase en un esquema One-vs-Rest y registra un AUC de 0.50 para las tres clases. Dicho valor, igual al área de la diagonal de referencia, indica que el modelo se comporta de forma equiparable a un clasificador aleatorio, sin capacidad real de discriminar entre una clase y las demás. La línea diagonal significa que el TPR (tasa de verdaderos positivos) crece a la misma velocidad que la FPR (tasa de falsos positivos), careciendo de una mejor o peor predicción que el mero azar.

La razón de este resultado es la configuración incorrecta del modelo: el uso de un MLPRegressor (en lugar de un MLPClassifier) con una tasa de aprendizaje excesiva provoca una convergencia inestable y una salida poco informativa, derivando en una clasificación casi aleatoria. La curva ROC, por tanto, confirma la ineficacia total del modelo para separar cada clase del resto de instancias.


Matriz de Confusión - Bad MLP (Test)
Real
Predicho
0
1
2
0
0
15
0
0
14
0
0
14
14
12
10
8
6
4
2
0

En esta matriz se ve con claridad que el modelo clasifica todos los ejemplos como la clase 2. Las filas correspondientes a las clases 0 y 1 muestran 0 aciertos y un número total de 15 y 14 errores, respectivamente. De igual forma, la clase 2 tiene 14 aciertos (predicción correcta de la misma clase) y 0 errores, lo que indica que el modelo aprendió a predecir prácticamente todo como clase 2. Este comportamiento confirma el pésimo desempeño del "Bad MLP", coherente con el 33% de precisión global y el AUC de 0.50 (equivalente a un clasificador aleatorio), resultado de un uso incorrecto de MLPRegressor y una tasa de aprendizaje excesiva.

Bono Shap Values:

Usamos Shap para comprender como cada una de las variables o características afectan la probabilidad predicha de que un país tenga un PIB alto, medio o bajo. Dado que tenemos un problema de clasificación con tres casos posibles (PIB alto, medio y bajo) ejecutamos tres gráficos de cascada, uno para cada caso posible. Veamos cuáles influyeron más y cuál es la intuición detrás del resultado obtenido.

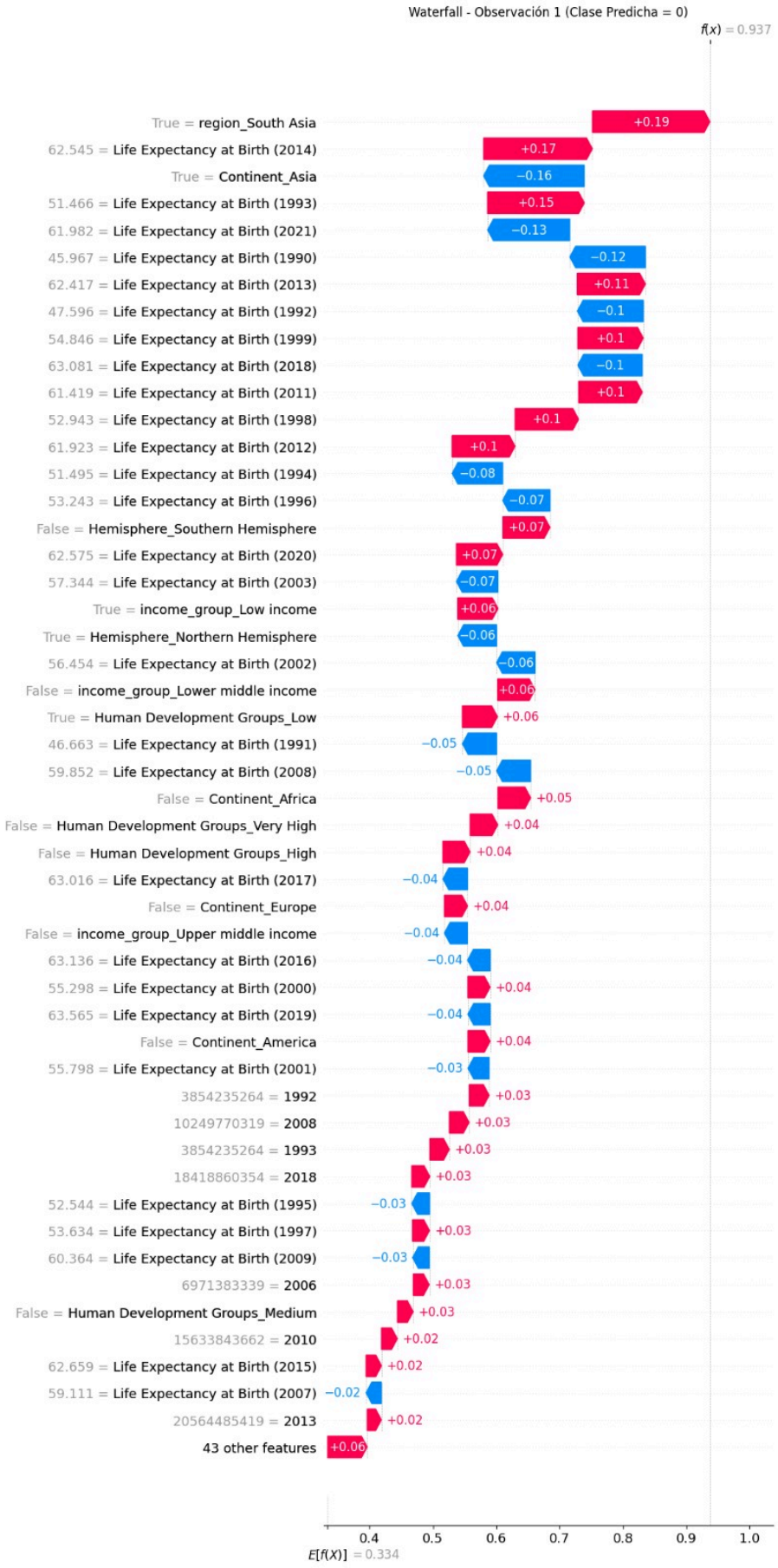

Waterfall - Observación 1 (Clase Predicha = 0)
f(x) = 0.937
True = region_South Asia +0.19
62.545 = Life Expectancy at Birth (2014) +0.17
True = Continent_Asia −0.16
51.466 = Life Expectancy at Birth (1993) +0.15
61.982 = Life Expectancy at Birth (2021) −0.13
45.967 = Life Expectancy at Birth (1990) −0.12
62.417 = Life Expectancy at Birth (2013) +0.11
47.596 = Life Expectancy at Birth (1992) −0.1
54.846 = Life Expectancy at Birth (1999) +0.1
63.081 = Life Expectancy at Birth (2018) −0.1
61.419 = Life Expectancy at Birth (2011) +0.1
52.943 = Life Expectancy at Birth (1998) +0.1
61.923 = Life Expectancy at Birth (2012) +0.1
51.495 = Life Expectancy at Birth (1994) −0.08
53.243 = Life Expectancy at Birth (1996) −0.07
False = Hemisphere_Southern Hemisphere +0.07
62.575 = Life Expectancy at Birth (2020) +0.07
57.344 = Life Expectancy at Birth (2003) −0.07
True = income_group_Low income +0.06
True = Hemisphere_Northern Hemisphere −0.06
56.454 = Life Expectancy at Birth (2002) −0.06
False = income_group_Lower middle income +0.06
True = Human Development Groups_Low +0.06
46.663 = Life Expectancy at Birth (1991) −0.05
59.852 = Life Expectancy at Birth (2008) −0.05
False = Continent_Africa +0.05
False = Human Development Groups_Very High +0.04
False = Human Development Groups_High +0.04
63.016 = Life Expectancy at Birth (2017) −0.04
False = Continent_Europe +0.04
False = income_group_Upper middle income −0.04
63.136 = Life Expectancy at Birth (2016) −0.04
55.298 = Life Expectancy at Birth (2000) +0.04
63.565 = Life Expectancy at Birth (2019) −0.04
False = Continent_America +0.04
55.798 = Life Expectancy at Birth (2001) −0.03
3854235264 = 1992 +0.03
10249770319 = 2008 +0.03
3854235264 = 1993 +0.03
18418860354 = 2018 +0.03
52.544 = Life Expectancy at Birth (1995) −0.03
53.634 = Life Expectancy at Birth (1997) +0.03
60.364 = Life Expectancy at Birth (2009) −0.03
6971383339 = 2006 +0.03
False = Human Development Groups_Medium +0.03
15633843662 = 2010 +0.02
62.659 = Life Expectancy at Birth (2015) +0.02
59.111 = Life Expectancy at Birth (2007) −0.02
20564485419 = 2013 +0.02
43 other features +0.06
0.4 0.5 0.6 0.7 0.8 0.9 1.0
E[f(X)] = 0.334

Gráfico de Cascada para PIB Bajo.

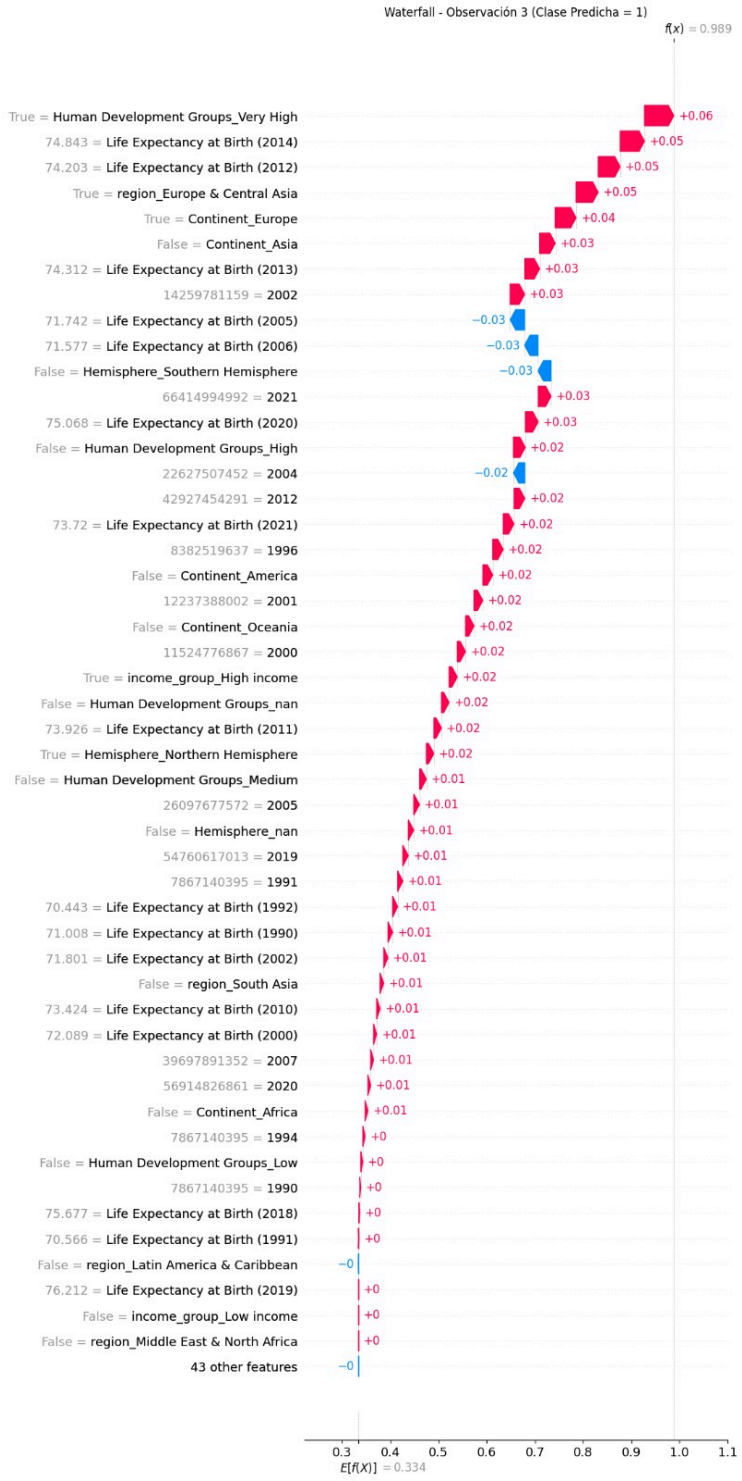


Gráfico de cascada para PIB Medio.

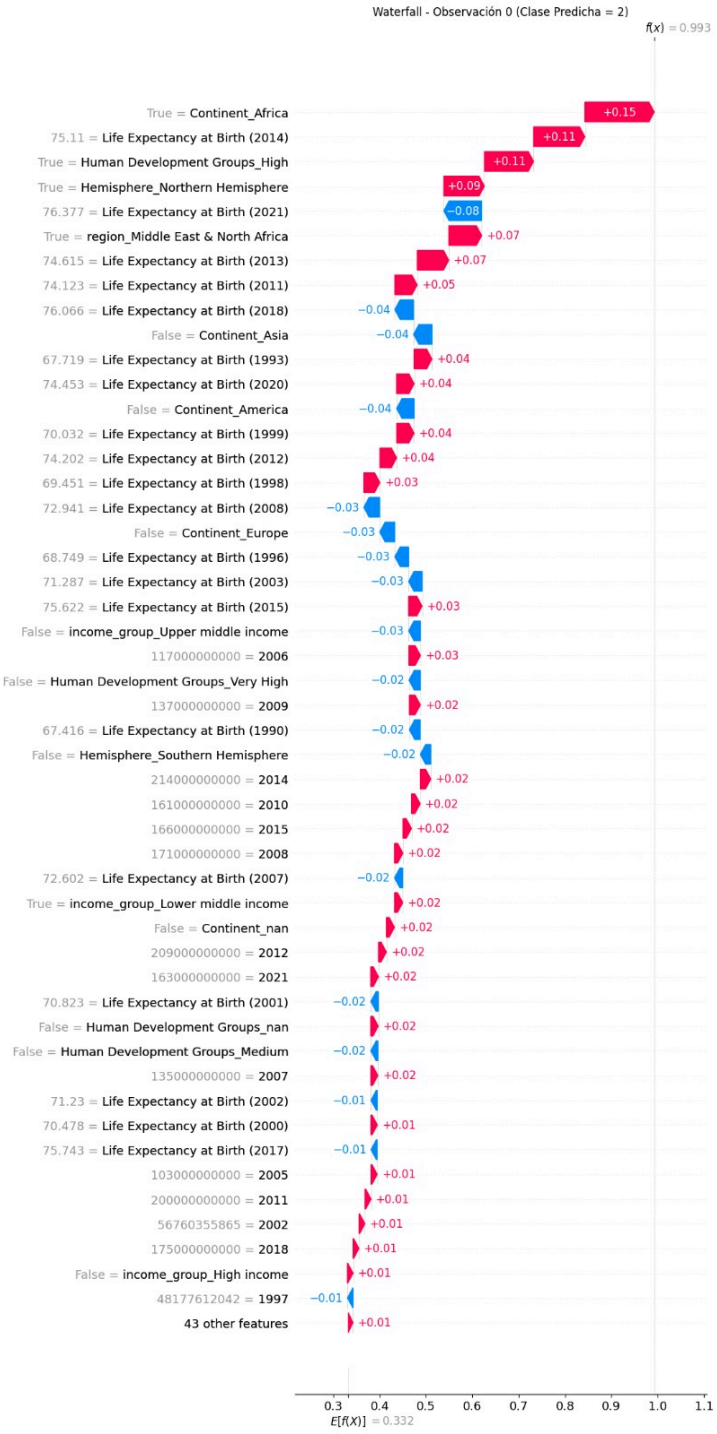


Gráfico de cascada para PIB Alto.

El primer gráfico de cascada nos muestra el aporte marginal de cada una de las variables con respecto al promedio predicho de todos los países para la categoría de PIB bajo. En este caso podemos ver que, en general, las características que más contribuyeron a la predicción son ser un país del sudeste asiático y la expectativa de vida del respectivo país. En este caso, la forma de interpretar el resultado es la siguiente*:*

La característica de ser un país asiático aumenta la probabilidad logarítmica predicha en 19 puntos porcentuales,, lo que significa que esta característica hace que sea más probable predecir que un país tenga un PIB bajo.

El segundo gráfico de cascada tiene una interpretación similar al primero, con la única diferencia de que el aporte marginal es respecto al promedio predicho de todos los países para la categoría de PIB medio. En este caso, se destaca la importancia de la característica del índice de desarrollo humano (IDH) representada dentro del gráfico del cascada como "Human Development Groups_Very High" y expectativas de vida. Finalmente, el tercer gráfico de cascada nos muestra la contribución marginal de cada una de las características con respecto al promedio predicho de todos los países para la predicción del PIB alto. En este caso, se resalta la importancia de pertenecer al continente Africano y, nuevamente, la expectativa de vida y el IDH. Finalmente no es posible asegurar que los valores otorgados por SHAP sean causales; esto debido a que los modelos de Machine Learning explotan la correlación entre variable para predecir las categorías. Es por esto que la herramienta de valores de SHAP debe verse como una herramienta que facilita la interpretabilidad de los modelos, más no una herramienta que nos otorga causalidad.