Reporte: Portafolio 2

Andre Sebastian Galindo Posadas A00833376

Inteligencia artificial avanzada para la ciencia de datos I (Gpo 102)

Profesor:

Jesús Adrián Rodríguez Rocha

Tecnológico de Monterrey

Sábado 7 de Septiembre 2024

Índice

Introducción

En el presente estudio, se utilizó un modelo de árbol de decisiones para analizar y predecir los resultados de las competencias olímpicas, específicamente, si un atleta o equipo ganará una medalla de oro, plata, bronce o ninguna. Para ello, se trabajó con un dataset que incluye información detallada de los Juegos Olímpicos, como los nombres de los competidores, su equipo, género, país, y el evento en el que participaron. A partir de estos datos, el objetivo principal fue identificar patrones que permitan predecir con precisión el desempeño de los participantes en futuras competiciones.

El árbol de decisiones se eligió como el modelo principal debido a su capacidad para manejar datos categóricos y numéricos, así como para interpretar de manera clara las reglas que conducen a las predicciones. No obstante, se enfrentaron varios desafíos relacionados con la complejidad del modelo, dado que el árbol generado presentó un gran número de niveles, nodos y hojas, lo que sugería una posible sobrecarga de información o overfitting. Para mejorar el rendimiento y evitar el sobreentrenamiento, se ajustaron varios hiper parámetros clave, como la profundidad máxima del árbol, el número mínimo de muestras por nodo y los criterios de partición.

A lo largo del análisis, se utilizaron diversas métricas de evaluación, incluyendo la precisión, el recall, el f1-score y el accuracy, las cuales mostraron resultados prometedores, con un promedio de precisión del 91.73%. Esto indicó que el modelo tenía un bajo sesgo y era capaz de capturar patrones importantes, aunque la elevada varianza sugirió la necesidad de optimizar su estructura para hacerlo más robusto. En consecuencia, se realizaron ajustes con el fin de reducir la complejidad del árbol, manteniendo al mismo tiempo un alto rendimiento predictivo.

Este estudio refleja el potencial de los modelos de árbol de decisiones para analizar datos deportivos y realizar predicciones útiles, al mismo tiempo que pone de manifiesto la importancia de controlar la complejidad del modelo para evitar problemas de sobreajuste y garantizar resultados confiables.

Problemática

En este proyecto, analizamos una base de datos de los Juegos Olímpicos con el objetivo de predecir si un atleta o un equipo ganará una medalla de oro, plata, bronce o no obtendrá medalla. Para ello, utilizamos un modelo de árbol de decisiones, una técnica de machine learning que nos permite identificar patrones en los datos después de un adecuado proceso de codificación y limpieza. Este enfoque nos ayuda a entender las relaciones entre diversas características de los competidores y sus resultados, permitiéndonos realizar predicciones sobre sus probabilidades de ganar una medalla.

Definición de Problema

En este problema, se utilizará un enfoque de Manching Learning, lo que implica la creación del árbol de Decisiones que pueda aprender patrones relevantes a partir de nuestros datos disponibles y, por lo tanto, predecir quiénes son los que ganan una medalla. Para esto el dataset será dividido en dos partes para el entrenamiento y las pruebas, la información que tenemos para los features y instances son, Sexo, Equipo, NOC, Año, Temporada, Ciudad, Deporte y Evemto. Y las medallas serán los Labels.

Origen de DataSet

En el caso de lo que se uso para el portafolio 2 es una base de datos de los juegos olimpicos, con toda la información relevante de los participantes y el titulo que ganaron en ese año, toda esta dataset fue sacado de kaggle en el siguiente link: https://www.kaggle.com/datasets/stefanydeoliveira/summer-olympics-medals-1896-2024

Datos y Limpieza

En este caso se mostrará la **Tabla 1** con los datos que tiene esta junto con el análisis de qué es lo relevante de esta base de datos y que es lo que se puede usar para la creación del modelo del árbol de decisiones

Nombre	Descripción	Tipo	Valores
Payer_id	Es el identificador único de cada jugador de los juegos olímpicos	Identificador/Entero	1,2,3..
Name	Nombre del participante de los juegos olímpicos	Textual	Strings
Sex	Sexo del jugador de los juegos olímpicos	Categórico	F, M
Team	Equipo con el que participo en los juegos olímpicos	Textual	Strings
NOC	Cómite nacional de deporte en el que está el jugador	Textual	Striings
Year	Año de los juegos olímpicos	Año/Entero	2002 2006 20012 2016
Seasson	Temporada de los juegos olímpicos	Textual	Strings
City	Lugar donde se celebran los juegos olímpicos	Textual	Strings
Sport	Deporte en el que participa el jugador	Textual	Strings
Event	Evento especifico en el que participo	Textual	Strings
Medal	Establece la posicion del jugador a travé de la medalla	Categórico	Oro, plata, bronce nada

Tabla 1. Son los datos extraídos del dataset, ya analizado, pero sin limpieza

Luego de haber visto la tabla procedemos con la limpieza y posteriormente con la asignación para el entrenamiento, ya que en este caso lo que se asigna como el feature e instances descartando ciertos parámetros son los siguientes. **Player_id** se descartó ya que simplemente es un identificador para cada jugador del dataset que realmente no da ninguna información importante para saber si este puede ganar o no una medalla. El otro dato que se descartó fue el **Name**, porque este es otro tipo de información que no aporta nada al modelo de los árboles de decisiones. Todo lo demás entra como un feature menos la **Medal,** por el hecho de que ese es nuestro Label que nos ayudará después para ver cómo se comporta el modelo. Este Dataset no contiene ningún Nan y todo estaba lleno. Además se usó el train_test_split para la separación de los datos de prueba y de entrenamiento del modelo, siendo los primeros los que se usan para después medir el modelo con las métricas correspondientes.

Modelo: Árbol de decisión

El árbol de decisiones se parece mucho a un árbol binario en el cual tienes nodos por los que vas bajando dependiendo de si cumples con alguna condición o patrón en particular, tienes los nodos hoja que son los que están hasta el final del árbol y el nodo raíz que es donde empieza. Para poder usar esta herramienta el dataset tuvo que ser codificado por el hecho de ser muchos datos Strings, por lo que se usó OneHotEncoder En este caso para la construcción del árbol lo primero que se hizo fue crear el modelo del árbol sin ningún hiper parámetro para

ver como resultaba este y luego ir ajustando la información para sacar el mejor modelo posible.

Las métricas con las que se evalúa son las siguientes:

- Accuracy (Exactitud): Esta métrica mide la proporción de predicciones correctas sobre el total de predicciones. En este caso, el accuracy refleja qué tan bien el modelo predice si un atleta o equipo ganará una medalla (oro, plata, bronce) o no. Un accuracy alto indica que el modelo tiene un buen desempeño general.

- Precision (Precisión): La precisión mide la proporción de verdaderos positivos (correctas predicciones de medallas) sobre el total de predicciones positivas. Esto indica qué tan confiable es el modelo cuando predice que se ganará una medalla, evitando falsos positivos (predecir una medalla cuando en realidad no se gana).

- F1-Score: El F1 es la media armónica entre precisión y recall, ofreciendo un balance entre ambas métricas. Es útil cuando hay una preocupación tanto por los falsos positivos como por los falsos negativos, especialmente en este contexto, donde predecir si alguien ganará o no una medalla tiene consecuencias importantes.

- Recall (Sensibilidad): El recall mide la proporción de verdaderos positivos sobre todos los casos que realmente son positivos (atletas o equipos que efectivamente ganaron medalla). Una alta sensibilidad implica que el modelo es capaz de detectar la mayoría de los ganadores de medallas, evitando falsos negativos.

El resultado del primer modelo donde no se usó ningún hiper parámetro, fue bastante alto ya que contó con los siguientes resultados en la **Tabla 2**:

Metrica/Resultado	Modelo 1(Sin Hiper parámetros)
Accuracy	0.9171
Presicion	0.9143
F1-Score	0.9153
Recall-Score	0.9171

Tabla 2: Se muestran los resultados del modelo que se creó sin usar los hiperparametros

El árbol de decisiones construido para el dataset de los Juegos Olímpicos mostró una complejidad notable con 232 niveles de profundidad, 38,425 nodos y 19,213 hojas. A pesar

de que las métricas obtenidas, como la precisión, f1-score, y recall, fueron bastante altas, con un promedio del 91.73%, esto indica que el modelo tiene un bias, ya que es capaz de hacer buenas predicciones basadas en los datos de entrenamiento.

Sin embargo, la varianza del modelo es considerablemente alta, lo que significa que el árbol podría estar capturando demasiado detalle del conjunto de entrenamiento, haciendo que sea más susceptible a cambios en los datos. Esto se refleja en la complejidad del árbol: su gran profundidad, número elevado de nodos y hojas sugieren un posible overfitting. Si bien las métricas sugieren que no ha alcanzado un nivel crítico de sobreentrenamiento, el tamaño del árbol no está respaldado por los resultados de las predicciones, lo que implica que el modelo podría estar ajustado en exceso.

Para abordar este problema y reducir la varianza, se alteraron varios hiper parámetros clave del modelo, con el fin de simplificar el árbol y evitar que capture demasiado detalle innecesario. Los hiper parámetros ajustados fueron:

- Profundidad máxima del árbol (max_depth): Limitar la cantidad de niveles en el árbol ayuda a controlar su complejidad y a evitar que se sobreajuste a los datos.

- Mínimo número de muestras por hoja (min_samples_leaf): Aumentar este valor obliga al árbol a generalizar más, reduciendo el número de nodos y evitando la creación de hojas con muy pocas muestras.

- Mínimo número de muestras para dividir un nodo (min_samples_split): Al aumentar este parámetro, el árbol necesita un mayor número de muestras antes de crear una nueva división, lo que evita divisiones innecesarias y contribuye a la simplificación del modelo.

- Criterio de partición (criterion): Al probar entre Gini y Entropía, se pueden ajustar los umbrales de clasificación y división, lo que también afecta la estructura del árbol.

El objetivo de ajustar estos hiper parámetros fue reducir la complejidad del árbol mientras se mantenía una buena precisión en las predicciones. Esto se logra mediante un proceso de

GridSearch, donde se evaluaron diferentes combinaciones de valores para encontrar el mejor balance entre bias y varianza, logrando un modelo más simple pero eficiente.

Lo primero que se hizo fue maximizar los hiperparametros y buscar complicar un poco más el árbol de decisiones en cuanto a su profundidad pero analizar si esto y las conexiones podían hacer que esto bajará y las métricas con el GridSearch, lanzaron estos hiperparametros y resultados:

- Hiper parámetros dados para bajar el éxito del modelo y evitar de esta manera el sobreentrenamiento

```python
param_grid = {
    'max_depth': [100, 150, 250],
    'min_samples_split': [75, 150, 225],
    'min_samples_leaf': [10, 50, 150],
    'criterion': ['gini', 'entropy']
}
```

Mejores parámetros: {'criterion': 'gini', 'max_depth': 100, 'min_samples_leaf': 10, 'min_samples_split': 75}

En la **Tabla 3** se puede ver los resultados del modelo pero con los hiperparametros anteriores

Metrica/Resultado	Modelo 2(Hiper parámetros Ajustados)
Accuracy	0.8723
Presicion	0.8442
F1-Score	0.8453
Recall-Score	0.8723

Tabla 3: Se muestran los resultados del modelo que se creó al ajustarlos hiperparametros

Con estos hiper parámetros se logró bajar todas las métricas del modelo pero sin dejar de ser un modelo sólido y confiable. Se perdió el Accuracy que se tenía anteriormente y se disminuyó a un 87.23%, pero con esto se puede evitar el sobreentrenamiento del árbol de decisiones, lo que bajó significativamente lo que era la varianza anterior y hizo que pasara de ser muy alta a ser media y el Bias se mantuvo siempre bajo

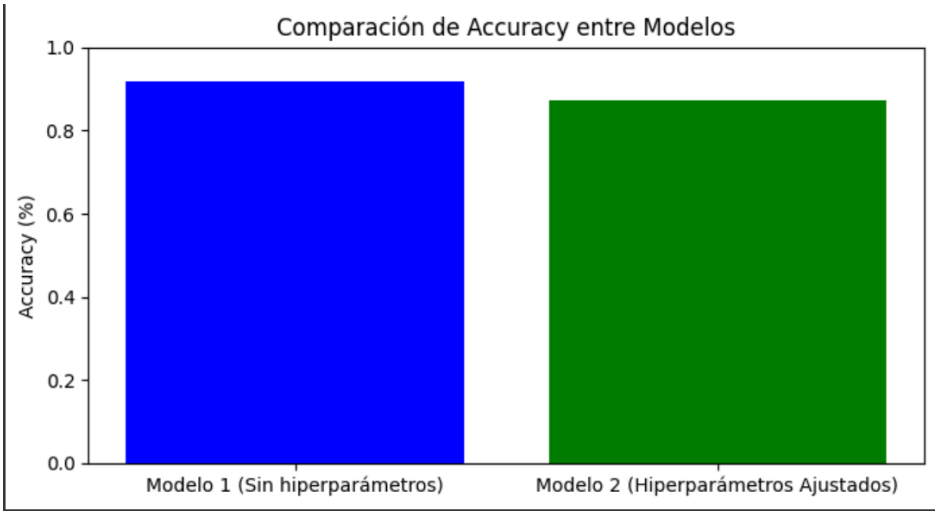
La precisión es de 84.42%, lo que indica que, aunque el modelo sigue siendo confiable cuando predice una medalla, ha bajado un poco en evitar falsos positivos. El f1-score también ha disminuido ligeramente a 84.53%, lo que sigue reflejando un buen equilibrio entre precisión y recall. Finalmente, el recall se mantiene alto en 87.23%, lo que significa que el modelo sigue detectando la mayoría de los casos correctos de medallas ganadas.

En la siguiente **Tabla 4** se ven los resultados finales y comparando los resultados de los dos modelos correspondientes

Metrica/Resultado	Modelo 1(Sin Hiper parámetros)	Modelo 2(Hiper parámetros Ajustados)
Accuracy	0.9171	0.8723
Presicion	0.9143	0.8442
F1-Score	0.9153	0.8453
Recall-Score	0.9171	0.8723

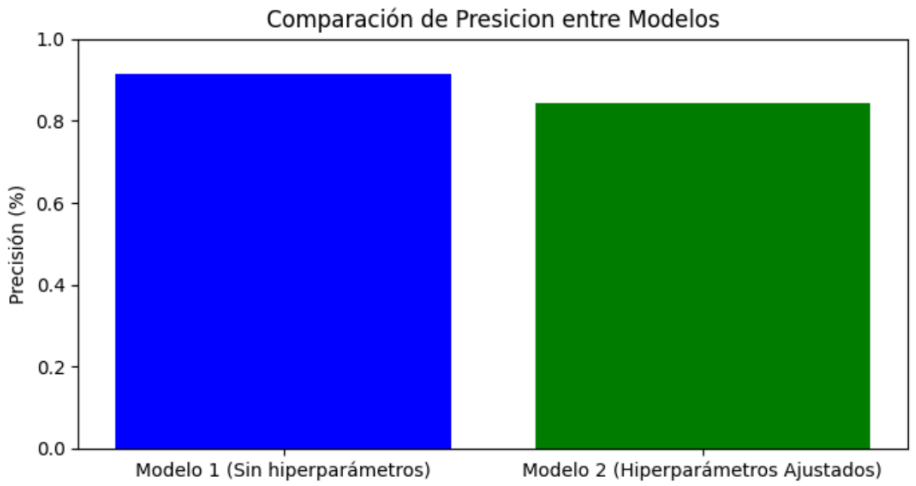
Tabla 4. Se ven los resultados y la comparación con el cambio de los hiper parámetros

En la **Gráfica 1** se ven los resultados comparados del Accuracy



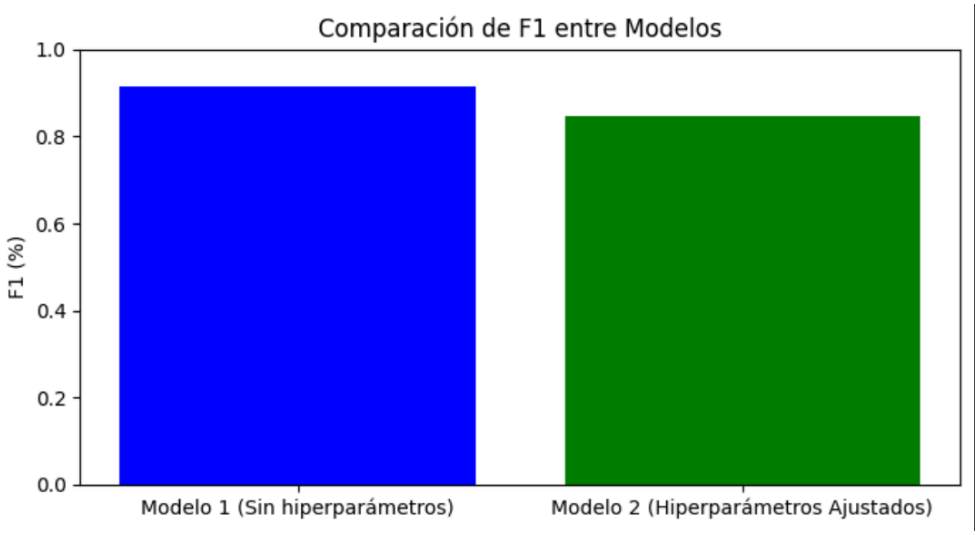
Gráfica 1: En esta gráfica se muestra cómo hubo un cambio en la Accuracy, luego de ajustar los hiper parámetros

En la **Gráfica 2** se ven los resultados comparados del Presicion



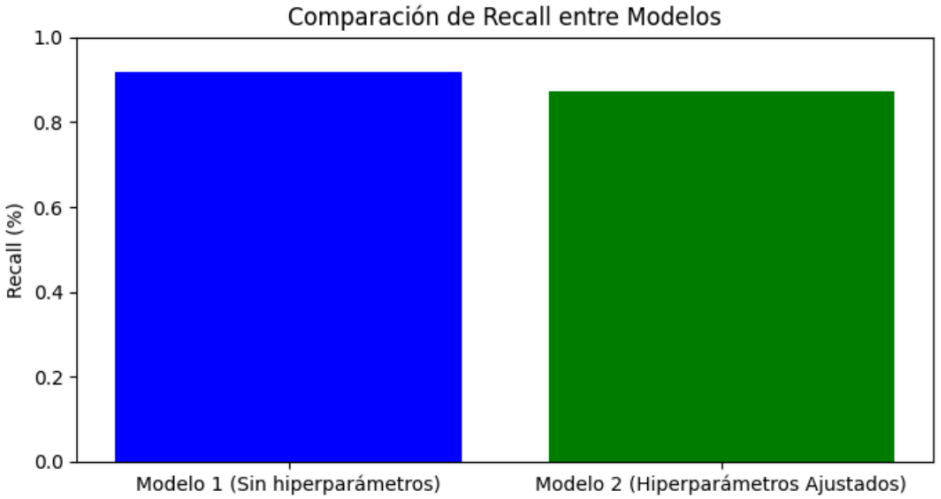
Gráfica 2: En esta gráfica se muestra cómo hubo un cambio en la Precisión, luego de ajustar los hiper parámetros

En la **Gráfica 3** se ven los resultados comparados del F1-Score



Gráfica 1: En esta gráfica se muestra cómo hubo un cambio en la F1-Score, luego de ajustar los hiper parámetros

En la **Gráfica 4** se ven los resultados comparados del Recall-Score



Gráfica 4: En esta gráfica se muestra cómo hubo un cambio en la Recall-Score, luego de ajustar los hiper parámetros

Conclusión

Con toda la información obtenida del árbol de decisiones, lo que se buscó en este caso fue reducir la complejidad del modelo para evitar el sobreentrenamiento. Inicialmente, el modelo era demasiado complejo, con una varianza muy alta y un bias o sesgo muy bajo. Tras los ajustes en los hiper parámetros, logramos que las métricas como la precisión y el f1-score disminuyeron ligeramente, pero se mantuvieran sólidas, lo que permitió que las predicciones siguieran siendo precisas. De este modo, evitamos el problema del sobreentrenamiento, ya que pasamos de tener una varianza alta a tener una de estado medio y un Bias bajo

Si bien este proceso puede parecer contrario a lo que normalmente se busca, ya que comúnmente tratamos de mejorar el modelo en lugar de "empeorarlo", esta situación demuestra que a veces es necesario limitar la capacidad del árbol de decisiones para que no genere un sobreajuste debido a su excesiva complejidad (alta Varianza, baja Bias). A pesar de la ligera disminución en las métricas, el modelo sigue siendo sólido y capaz de hacer predicciones precisas (media Varianza, bajo Bias).

Referencias Bibliográficas

- *Tutorial de clasificación en árbol de decisión en Python*. (2024, enero). Datacamp. Recuperado 7 de septiembre de 2024, de https://www.datacamp.com/es/tutorial/decision-tree-classification-python?utm_source=google&utm_medium=paid_search&utm_campaignid=20616617505&utm_adgroupid=154290358037&utm_device=c&utm_keyword=&utm_matchtype=&utm_network=g&utm_adpostion=&utm_creative=711801245249&utm_targetid=aud-517318241987:dsa-2220216603507&utm_loc_interest_ms=&utm_loc_physical_ms=1010132&utm_content=&utm_campaign=231025_1-sea~dsa~tofu_2-b2c_3-es-lang_4-prw_5-na_6-na_7-le_8-pdsh-go_9-nb-s_10-na_11-na-sep24&gad_source=1&gclid=Cj0KCQjw8--2BhCHARIsAF_w1gzL8memepZqDjgszdjSGn-hE_TlkREnHL94oE-2UHwfE28kn4NJiggaAguEEALw_wcB
- *Summer Olympics medals (1896-2024)*. (2024, 29 agosto). Kaggle. https://www.kaggle.com/datasets/stefanydeoliveira/summer-olympics-medals-1896-2024
- Navarro, S. (2024, 18 abril). Grid search en Python | KeepCoding Bootcamps. *KeepCoding Bootcamps*. https://keepcoding.io/blog/grid-search-en-python/
- *Python Data Visualization (with examples) | Hex*. (s. f.). https://hex.tech/use-cases/data-visualization/python-visualization/?utm_id=h_701VU0000099OFdYAM&utm_source=youtube_ads&utm_medium=paid_social&utm_campaign=19789284653&utm_content=153592932384&gclid=Cj0KCQjw8--2BhCHARIsAF_w1gzUjawkdNBfDzR6gWorp0gdFKn0RgThjijWgZZ12Sk4LzBYlcs2NgAaAnhvEALw_wcB
-