**ÍNDICE**

[1. INTRODUCCIÓN 1](#_Toc183549001)

[2. JUSTIFICACIÓN 2](#_Toc183549002)

[3. VALIDACIÓN TÉCNICA 3](#_Toc183549003)

[3.1 Comprensión del negocio 3](#_Toc183549004)

[3.2 Comprensión de los datos 3](#_Toc183549005)

[3.3 Preparación de los datos 7](#_Toc183549006)

[3.4 Modelado 9](#_Toc183549007)

[3.4.1 Arboles 9](#_Toc183549008)

[3.4.2 Naives 10](#_Toc183549009)

[3.4.3 Regresión lineal 12](#_Toc183549010)

[3.4.4 Redes neuronales 15](#_Toc183549011)

[3.5 El modelo 18](#_Toc183549012)

[3.6 Evaluación 20](#_Toc183549013)

[4. Interpretación de los resultados 22](#_Toc183549014)

[4.1 Conclusiones 22](#_Toc183549015)

[4.2 Recomendaciones 22](#_Toc183549016)

# INTRODUCCIÓN

Con el objetivo de optimizar su algoritmo de predicción de resultados en la UEFA Champions League, la casa de apuestas Codere solicitó la colaboración de nuestro equipo para analizar patrones en los equipos participantes desde el año 2010. Este análisis busca evaluar la confiabilidad de las predicciones generadas por su sistema, con la intención de fortalecer la confianza de los usuarios y promover el uso de los servicios ofrecidos por la empresa.

# JUSTIFICACIÓN

Este estudio tiene como objetivo realizar un análisis descriptivo para identificar los factores que han influido en el éxito de los equipos en la UEFA Champions League desde 2010. Al examinar las características de los jugadores y los detalles de los partidos, buscamos detectar patrones que expliquen el desempeño exitoso de ciertos equipos. Estos hallazgos no solo permitirán mejorar la precisión de las predicciones realizadas por Codere, sino que también ofrecerán una base sólida para comprender las dinámicas del fútbol de alto nivel.

Para llevar a cabo este análisis, se emplearon algoritmos desarrollados en Python, los cuales permitieron explorar en profundidad los patrones identificados. Las técnicas utilizadas incluyen:

* Árboles de decisión, para clasificar y prever comportamientos basados en criterios jerárquicos.
* Método de Bayes, para calcular probabilidades condicionadas en función de las características observadas.
* Regresión lineal, para identificar relaciones entre variables clave y resultados en los partidos.

El enfoque descriptivo no solo facilitará la optimización del algoritmo de predicción de Codere, sino que también fortalecerá la confianza en sus servicios y atraerá a más usuarios interesados en aprovechar estas herramientas avanzadas.

# VALIDACIÓN TÉCNICA

Para garantizar la precisión y confiabilidad de los resultados obtenidos, se llevaron a cabo diversos procesos, incluyendo la obtención de datos de fuentes confiables. Entre estas destacan Kaggle, una plataforma de competencia en ciencia de datos, y Transfermarkt, un portal especializado en fútbol que ofrece información sobre fichajes, valores de mercado, rumores, estadísticas, noticias de transferencias y partidos. A partir de estas fuentes de información, se extrajeron los datos necesarios y se procesaron con el propósito de realizar predicciones y buscar patrones relevantes.

## Comprensión del negocio

Codere es una empresa multinacional española dedicada al sector del juego y las apuestas. Fundada en 1980, se ha consolidado como uno de los principales operadores de juegos de azar en países de Europa y América Latina. Su modelo de negocio incluye la gestión de casinos, salas de bingo, máquinas recreativas y apuestas deportivas, tanto en establecimientos físicos como a través de plataformas en línea.

En el ámbito de las apuestas deportivas, Codere ofrece una amplia gama de servicios que incluyen predicciones de eventos deportivos, en particular fútbol, basadas en estadísticas y análisis detallados. Además, es reconocida por ser patrocinadora de equipos y eventos deportivos, lo que refuerza su presencia en el mercado.

Codere se destaca por su enfoque en la innovación tecnológica, empleando herramientas avanzadas de análisis y predicción para mejorar la experiencia del usuario. Su objetivo principal es brindar servicios confiables, atractivos y alineados con las normativas legales de cada región en la que opera.

## Comprensión de los datos

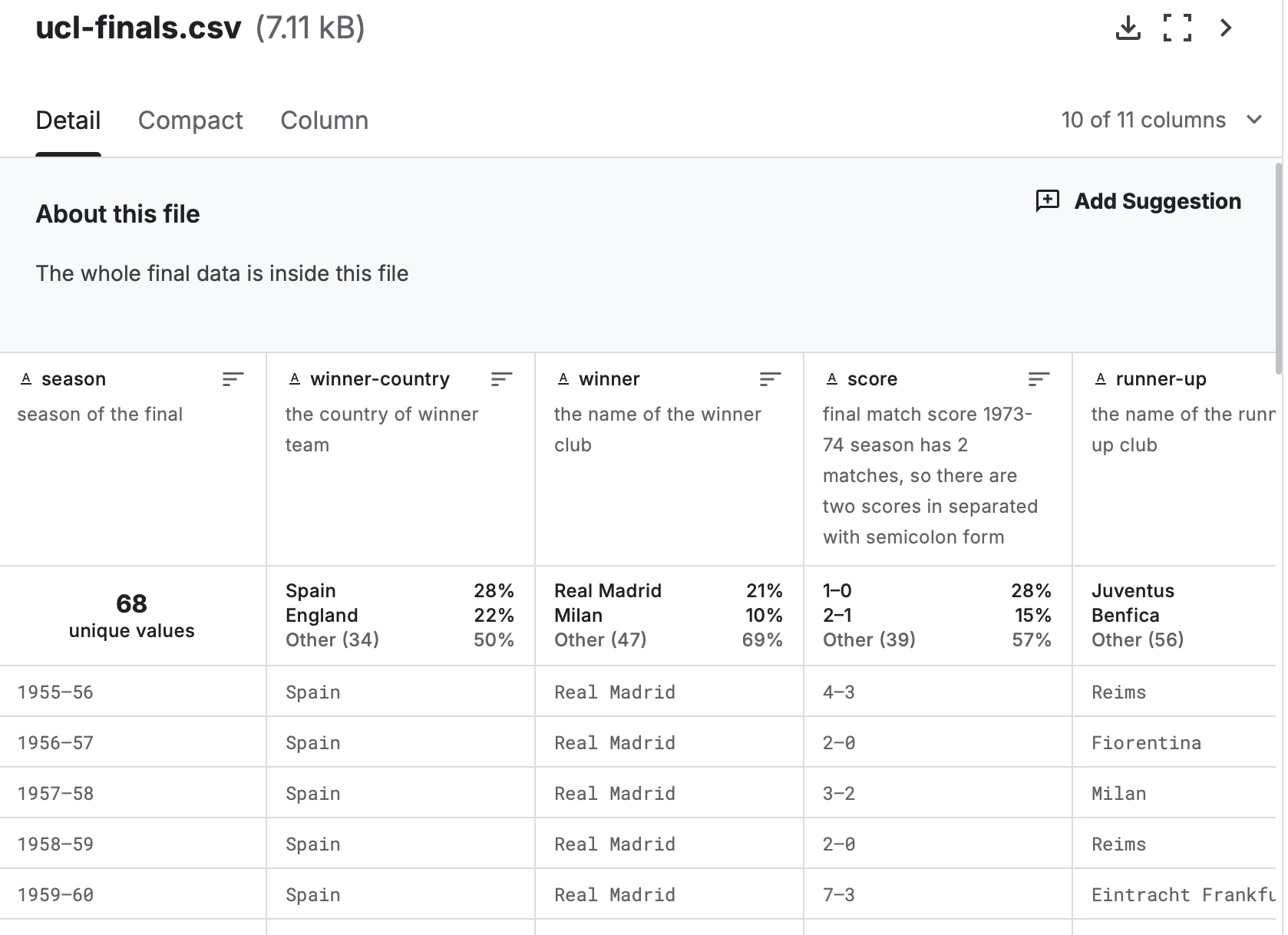
Los datos de jugadores y equipos provienen de fuentes confiables, incluyendo estadísticas oficiales de la UEFA y otras bases de datos deportivas. La calidad de los datos fue verificada mediante un proceso de limpieza y estandarización, asegurando la precisión en los nombres de los jugadores, posiciones, edades y detalles de cada final. La revisión de estos datos es fundamental para garantizar un análisis sólido y confiable.

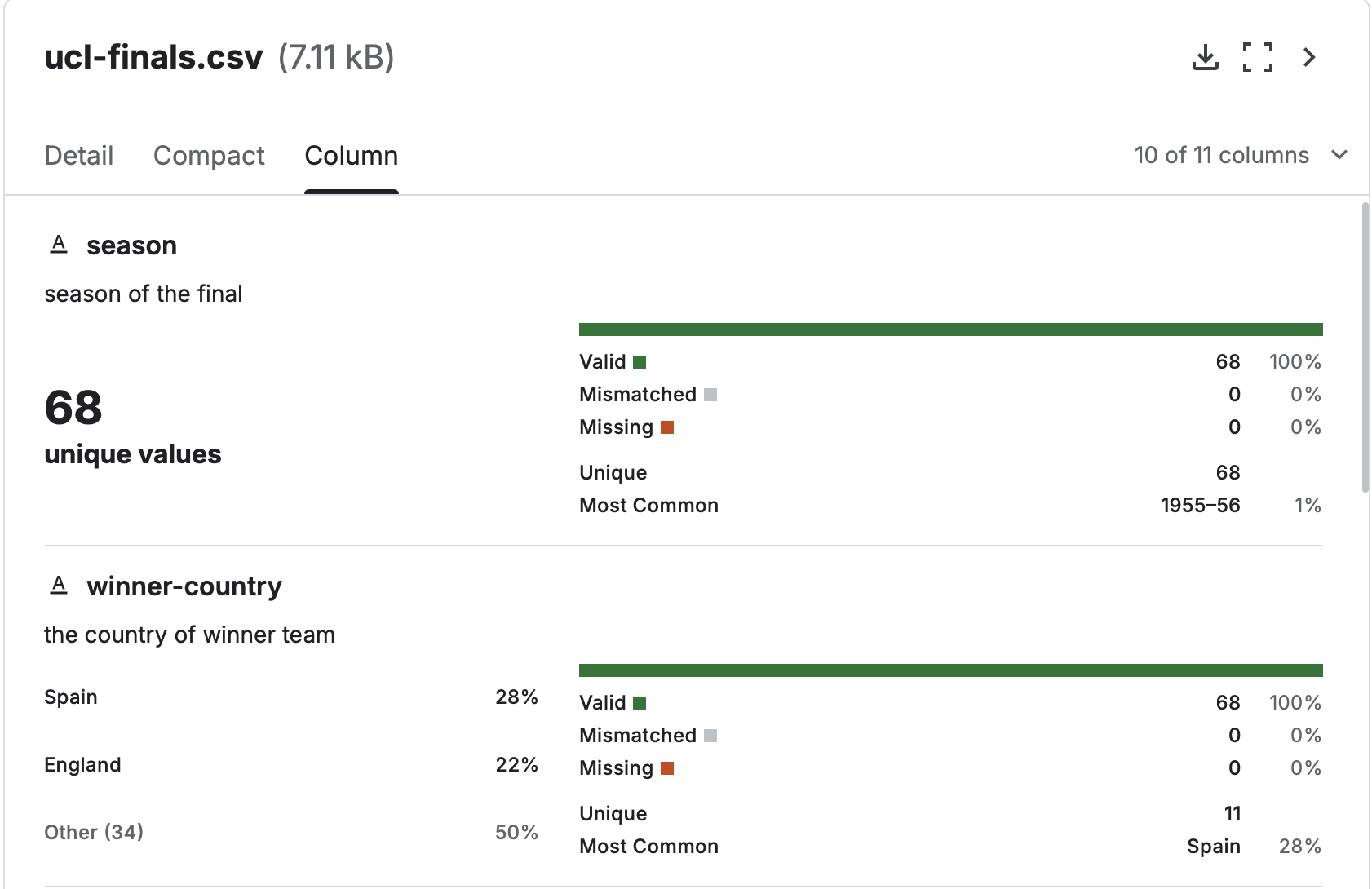
Uno de los datasets fue obtenido de **Kaggle.com**, una plataforma que ofrece una amplia variedad de datasets utilizados principalmente para minería de datos debido a su gran volumen.

Dicho dataset incluye los siguientes campos:

* **Temporada**: Contiene los años de la temporada, por ejemplo, la temporada 2024-2025.
* **País del ganador**: Indica el país de origen del equipo ganador del partido de la final.
* **Ganador**: Nombre del equipo que ganó el partido.
* **Marcador**: Resultado final del partido, incluyendo los 90 minutos y el tiempo extra, si se llegó a dicha fase.
* **Subcampeón**: Nombre del equipo derrotado en la final.
* **País del subcampeón**: País de origen del equipo subcampeón.
* **Estadio**: Nombre del estadio donde se jugó la final.
* **Ciudad de la final**: Ciudad donde tuvo lugar el partido.
* **País de la final**: País en el que se ubica el estadio.
* **Asistencia**: Número de asistentes al partido en el estadio.
* **Forma de ganar**: Especifica si el partido se decidió en los 90 minutos, en tiempo extra o en tanda de penales.

En las siguientes imágenes se presentan análisis estadísticos de los campos, destacando los valores más comunes, la cantidad de datos únicos y los porcentajes de los registros.

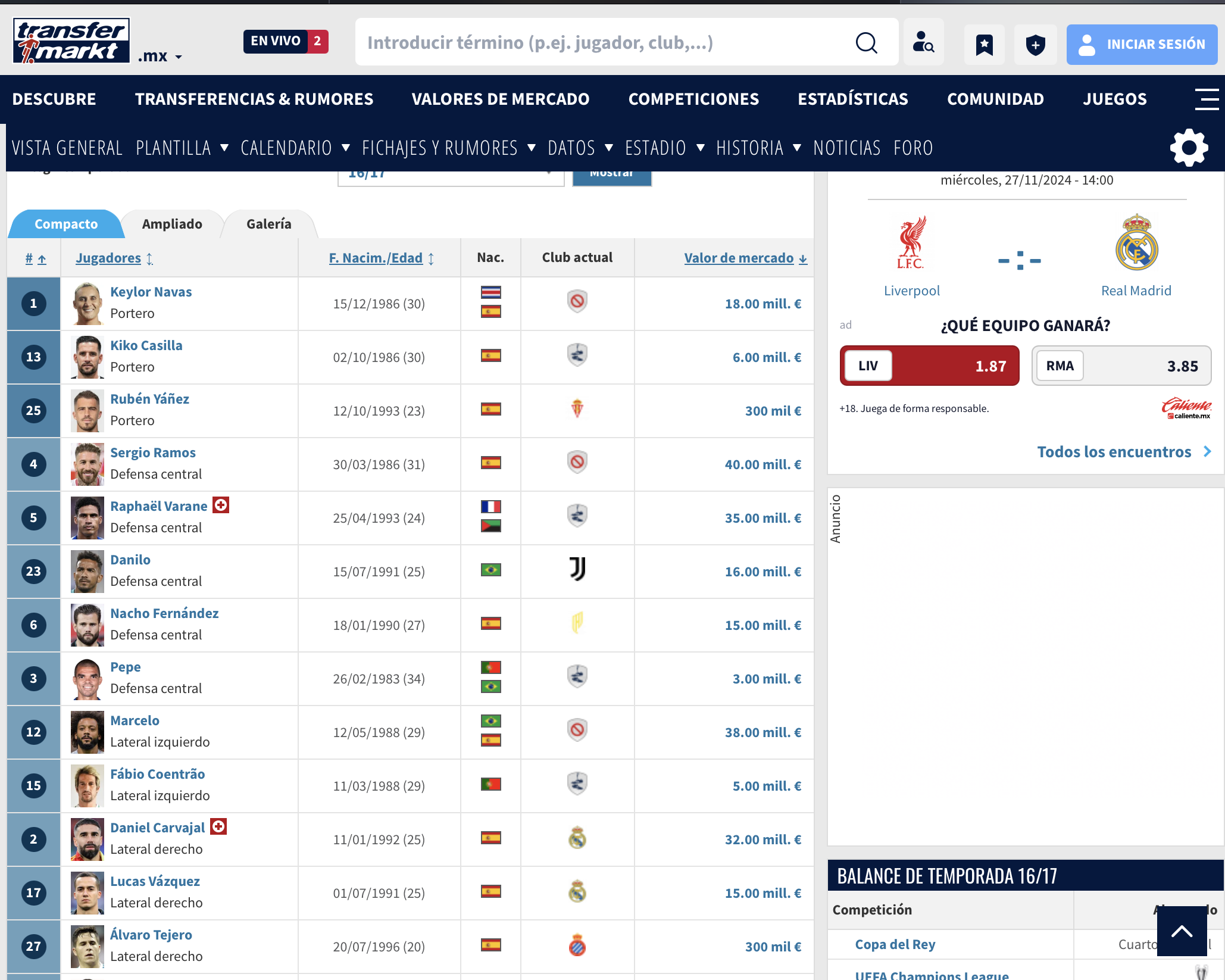


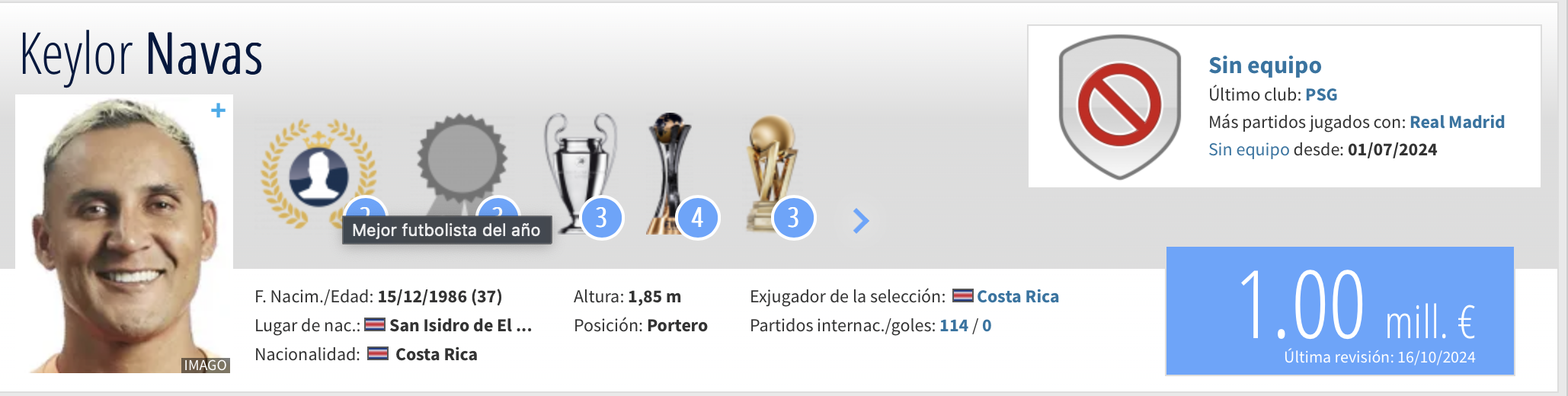


Por otro lado, el segundo dataset fue creado íntegramente por nosotros con el objetivo de adaptarlo a las necesidades del análisis y las predicciones requeridas. Para su construcción, utilizamos información del primer dataset y buscamos detalles adicionales sobre los equipos involucrados en los partidos, como los jugadores que formaron parte del plantel en esa temporada. Este dataset incluye los siguientes campos:

* **Año**: Representa el año de inicio de la temporada, para vincular este dataset con el primero.
* **Equipo**: Nombre del equipo participante en el partido.
* **Nombre**: Nombre del jugador que estuvo en el plantel durante esa temporada.
* **Nacionalidad**: País al que representa el jugador. En caso de doble nacionalidad, se seleccionó la nacionalidad con la que el jugador decidió representar oficialmente.
* **Edad**: Edad del jugador en esa temporada.
* **Posición**: Rol desempeñado por el jugador en el partido, como portero, defensa, delantero, etc.

En las siguientes imágenes se muestra cómo el portal **Transfermarkt** presenta los datos.





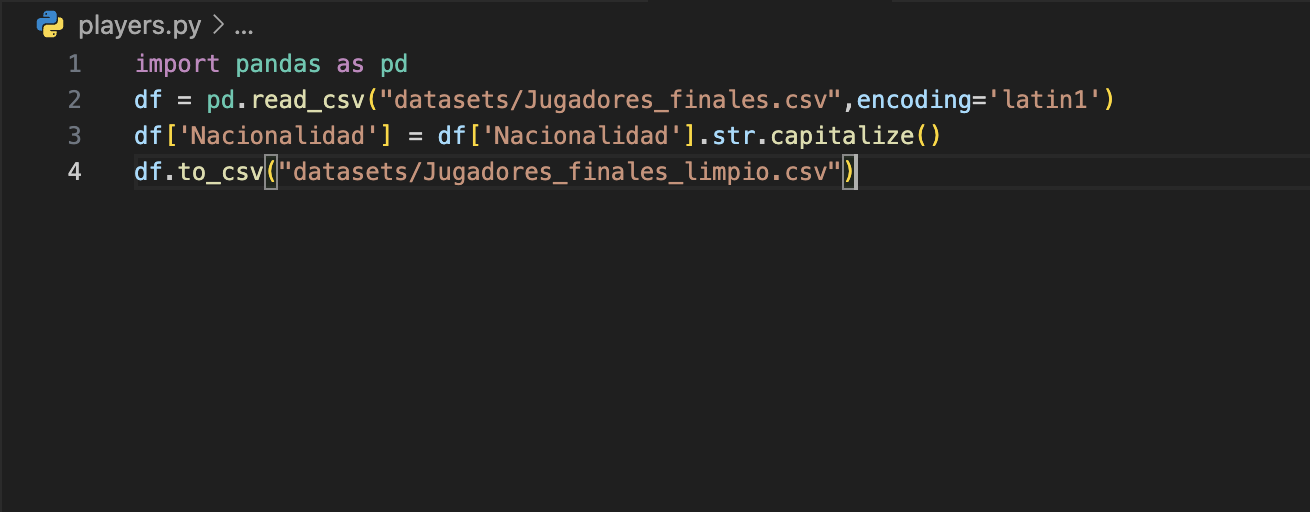
## Preparación de los datos

Para el correcto procesamiento de los datos, fue fundamental realizar una **limpieza** exhaustiva. Este proceso incluyó la corrección en la escritura de los nombres de los equipos y los campos de los datasets, la traducción de los nombres de los países y la eliminación de acentos. Estas acciones permitieron una mejor comprensión y normalización de los datos por parte del sistema.

Posteriormente, ambos datasets fueron unificados en uno solo para facilitar su análisis y la aplicación de los diferentes modelos previamente mencionados.

En las siguientes líneas de código, desarrolladas en el lenguaje Python, se pueden observar los procesos llevados a cabo para lograr esta limpieza y unificación.







## Modelado

El problema principal a resolver es identificar y predecir los factores clave que determinan el éxito de los equipos en la UEFA Champions League, utilizando los datos históricos recolectados. Para ello, se implementaron modelos que analizan patrones en el desempeño de equipos y jugadores, permitiendo generar predicciones precisas sobre los resultados de futuros partidos y finales.

**Variables del dataset**

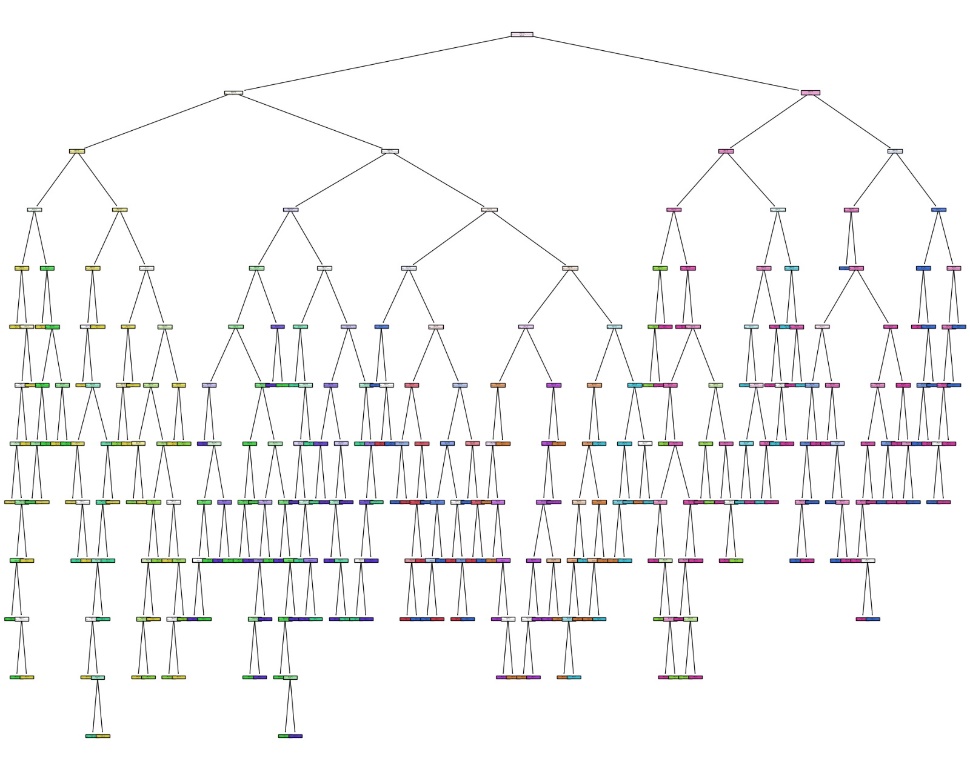
El dataset contiene información sobre finales de la UEFA Champions League. Las variables principales son:

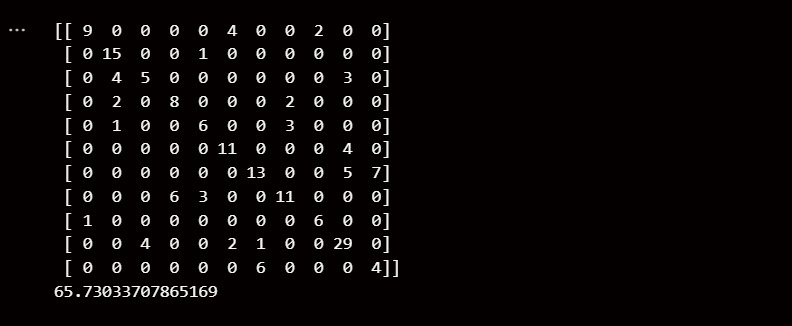
1. **Año:** Año en que se jugó la final.
2. **Nombre:** Nombre del jugador o individuo registrado.
3. **Equipo:** Equipo participante.
4. **Nacionalidad:** Nacionalidad de los jugadores o equipos.
5. **Edad:** Edad de los jugadores (si aplica).
6. **Posición:** Posición de los jugadores (si aplica).
7. **Season:** Temporada de la competencia.
8. **Winner-country:** País del equipo ganador.
9. **Score:** Marcador del partido.
10. **Runner-up:** Equipo subcampeón.
11. **Runner-up-country:** País del equipo subcampeón.
12. **Stadium:** Estadio donde se jugó la final.
13. **Final-city:** Ciudad de la final.
14. **Final-country:** País de la final.
15. **Attendance:** Asistencia registrada en el estadio.
16. **Winning-way:** Forma en que ganó el equipo (tiempo regular, penales, etc.).
17. **Year:** Año (variable redundante con Año).
18. **Winner\_score:** Goles anotados por el equipo ganador.
19. **Runner\_up\_score:** Goles anotados por el equipo subcampeón.
20. **Winner:** Equipo ganador (variable objetivo).

## Arboles

Utilizando el método de arboles se puede observar con la siguiente imagen que nos arroja una gran cantidad de ramas por lo que lo ideal seria podarlo, sin embargo, no es posible en estos momentos.

Con el modelo tal como se muestra se obtuvo que con este sistema es fiable en un 65.73% por lo que no es algo viable para nuestro uso





Existen diferentes tecnicas de poda para casos como este donde el modelo retorna una gran cantidad de ramas, entre las cuales destacan las siguientes:

1. Poda previa (pre-pruning):

* En esta técnica, se limitan las características del árbol antes de que se construya completamente.
* Algunos parámetros comunes utilizados son:
  + max\_depth: Define la profundidad máxima del árbol.
  + min\_samples\_split: Número mínimo de muestras necesarias para dividir un nodo.
  + min\_samples\_leaf: Número mínimo de muestras que debe tener una hoja.
  + max\_leaf\_nodes: Número máximo de hojas permitidas en el árbol.

1. Poda posterior (post-pruning):

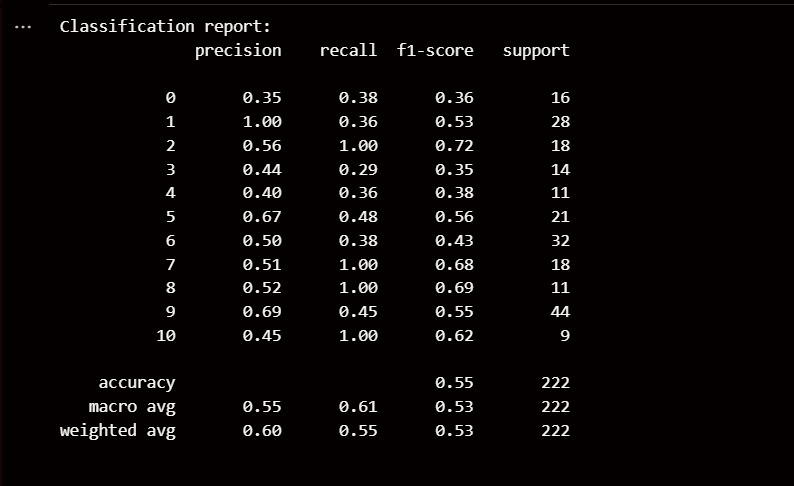
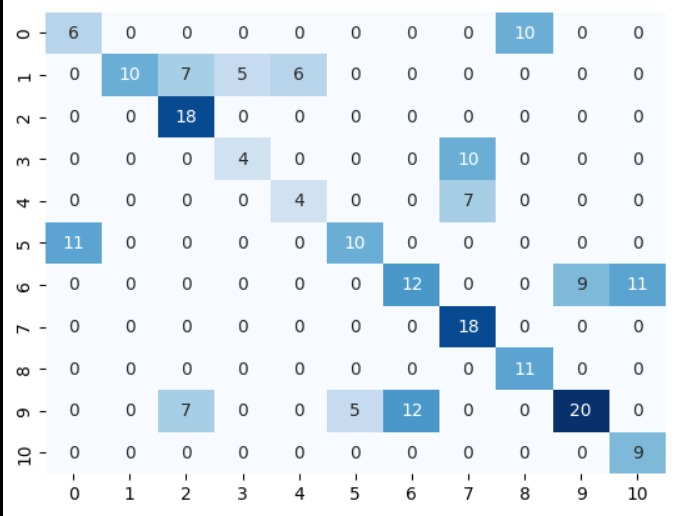
* Esta técnica implica construir primero el árbol completo y luego eliminar ramas no significativas.
* En scikit-learn, se utiliza el parámetro ccp\_alpha (Costo Complejidad de Poda) para realizar esta operación.
* El proceso implica seleccionar un valor de ccp\_alpha que balancee el tamaño del árbol y su desempeño.

### Naives

Las variables utilizadas para este modelo son las ya definidas en el punto 3.4, para este modelo se utilizan cada una de estas variables al realizar la predicción.

Se inicia eliminando la tabla equipo y procedemos a definir “X” desde la primera posición de los datos hasta la última, analizando todas las columnas que este contiene. Se define a “Y” como la variable a predecir, siendo la columna equipo. Una vez declarados estos datos se procede a entrenar el modelo con el tren de aprendizaje, dándole un 75% de los datos para entrenar el modelo y dejando el resto para la evaluación.

Mediante Naives se obtuvo un reporte de clasificación con una precisión del 55% analizando cada uno de los 10 resultados de las finales, siendo esta una precisión muy baja por lo que no es recomendable utilizar este modelo.

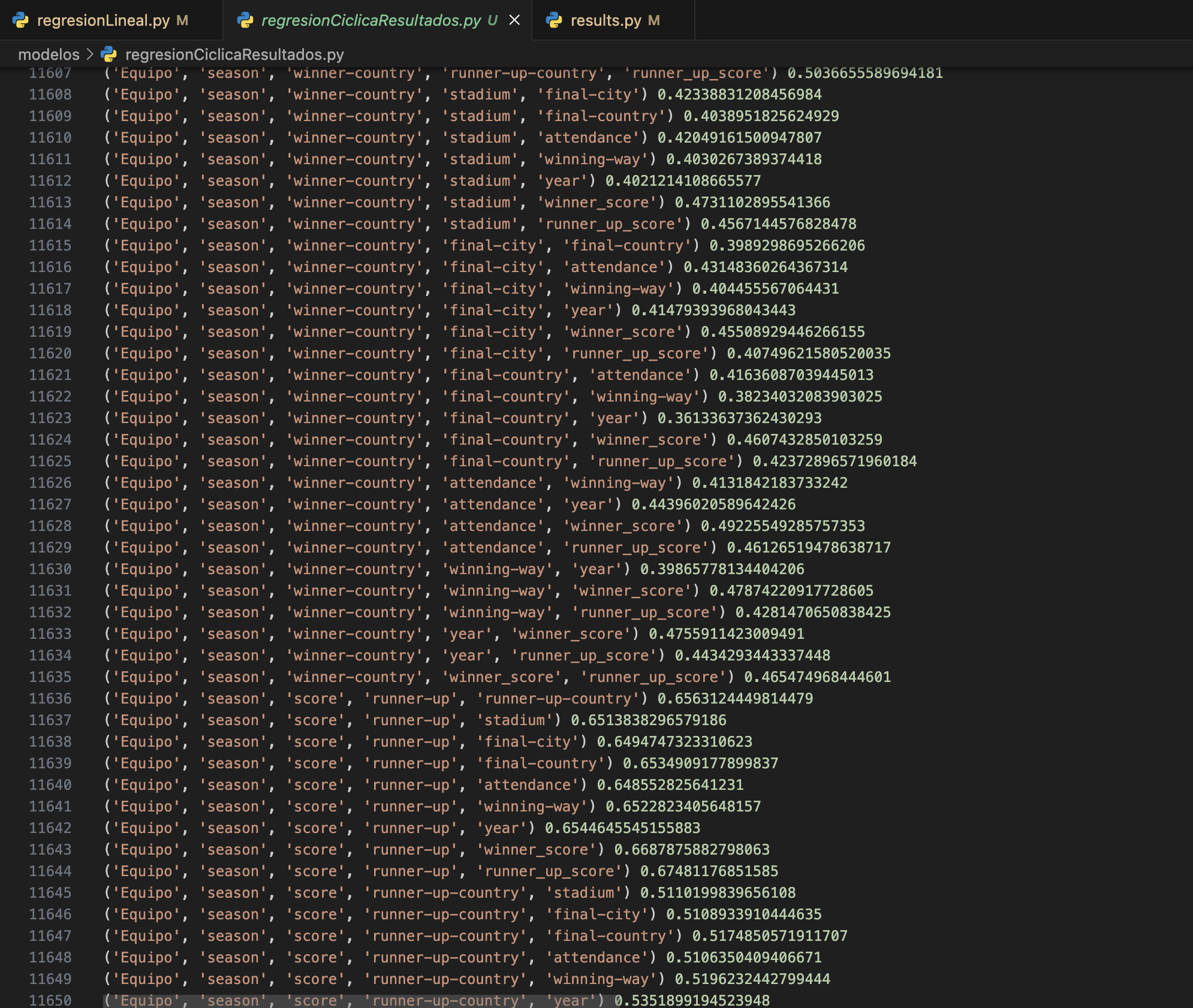
** **

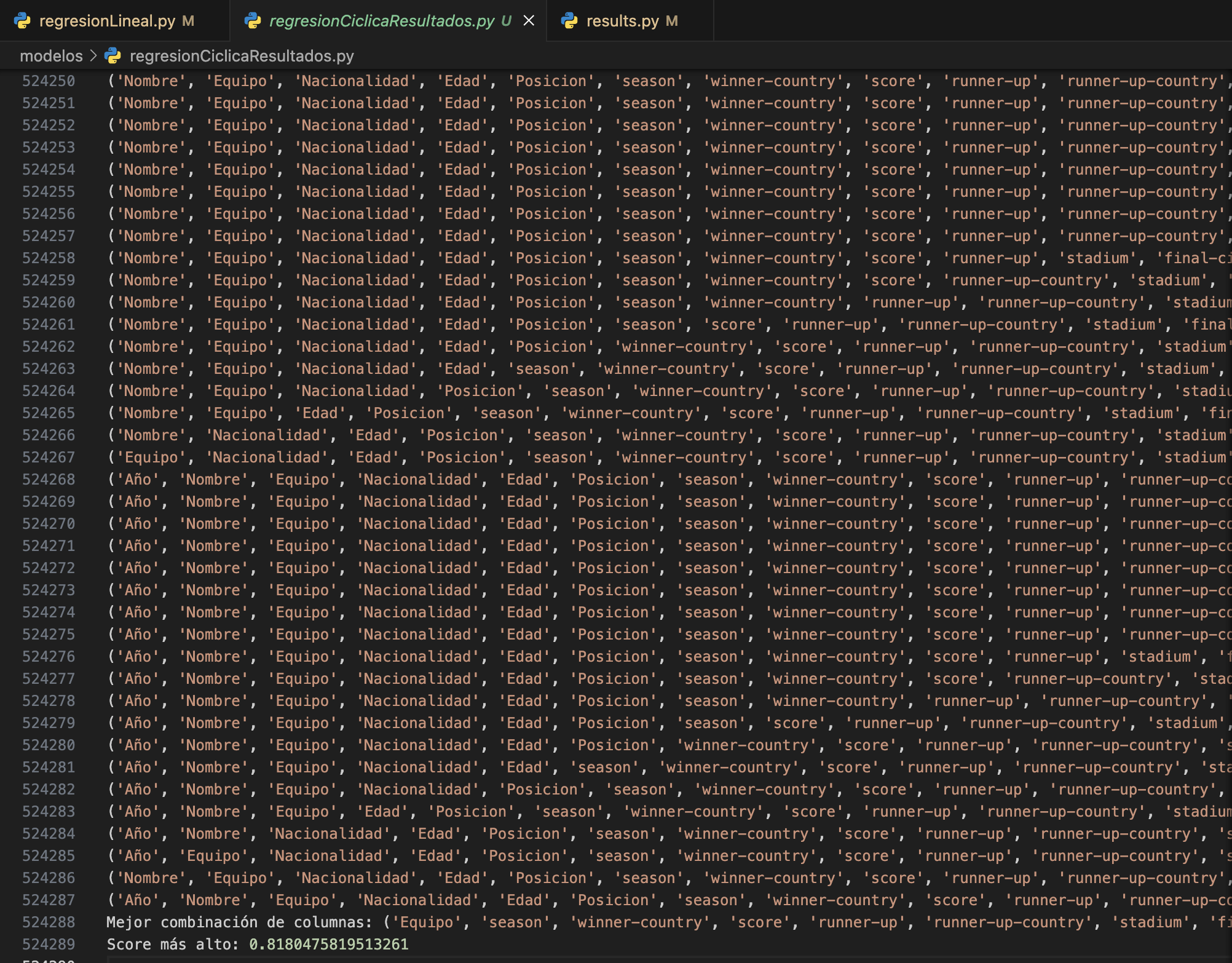
### Regresión lineal

Mediante este modelo se observo mediante un mapa de calor la influencia de los diferentes campos que tenemos en nuestro dataset en función de la capacidad de predecir los datos, mientras mas rojos esten signfica que mayor correlación tiene con el resto de los campos.



Con diferentes algoritmos capaces de analizar las distintas combinaciones posibles y obtener el nivel de confianza de cada iteración dando los siguientes resultados:





Sin embargo, dado que arrojo un total de 524,287 resultados, se opto por la construcción de otro algoritmo mas el cual diera los 5 mejores y 5 peores resultados dando la ejecución del mismo lo siguiente:

Top 5 combinaciones:

Combinación: ('Equipo', 'season', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'year', 'winner\_score', 'runner\_up\_score'), Score: 0.8180475819513261

Combinación: ('Año', 'Equipo', 'season', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'winner\_score', 'runner\_up\_score'), Score: 0.8169714365603514

Combinación: ('Año', 'Equipo', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'year', 'winner\_score', 'runner\_up\_score'), Score: 0.8169714365603514

Combinación: ('Año', 'Equipo', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'winner\_score', 'runner\_up\_score'), Score: 0.8169714365603513

Combinación: ('Equipo', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'year', 'winner\_score', 'runner\_up\_score'), Score: 0.8169714365603511

Bottom 5 combinaciones:

Combinación: ('Nombre', 'Edad', 'stadium', 'attendance', 'runner\_up\_score'), Score: -0.09951826190984625

Combinación: ('Nombre', 'Posicion', 'stadium', 'attendance', 'runner\_up\_score'), Score: -0.09984684761114271

Combinación: ('Nombre', 'stadium', 'attendance', 'runner\_up\_score'), Score: -0.09991516060204608

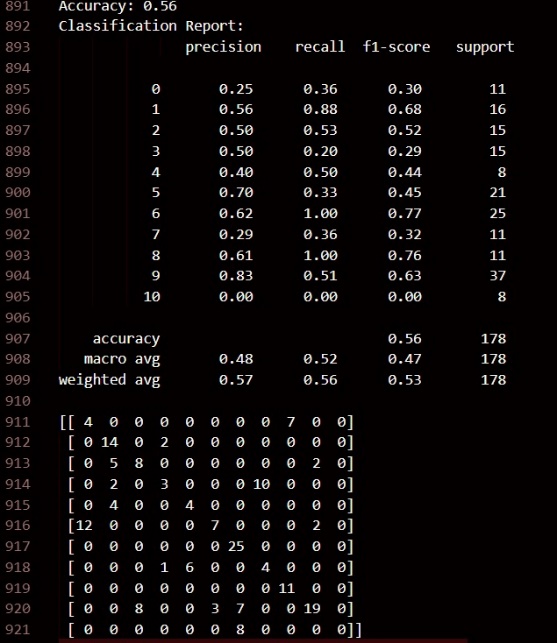
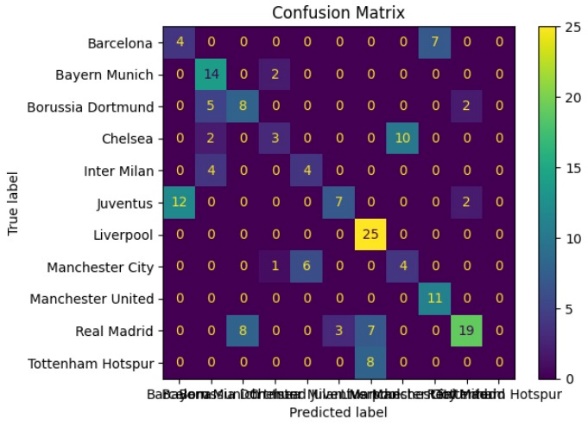
Combinación: ('Equipo', 'season', 'final-city', 'year'), Score: -0.42993839662608635

Combinación: ('Equipo', 'season', 'attendance', 'year'), Score: -0.4872241286214076

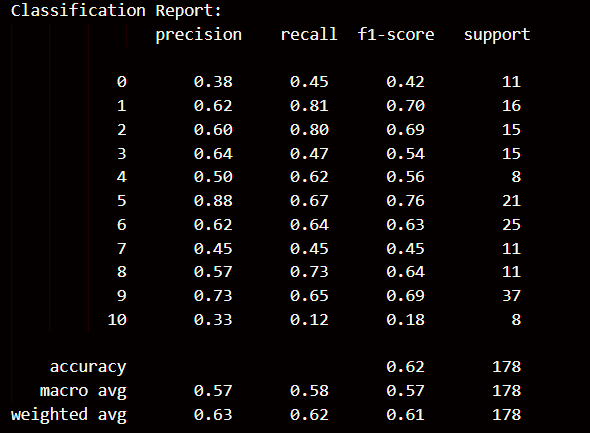
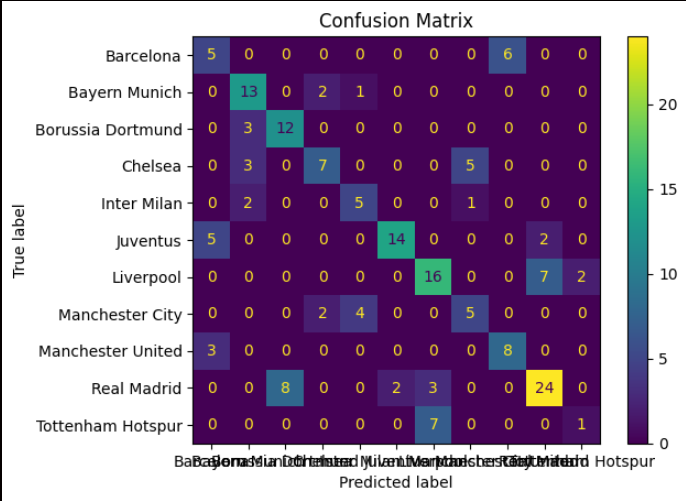
### Redes neuronales

Se realizan varias combinaciones y diferentes valores en las neuronas, creando resultados distintos para cada variación utilizada.

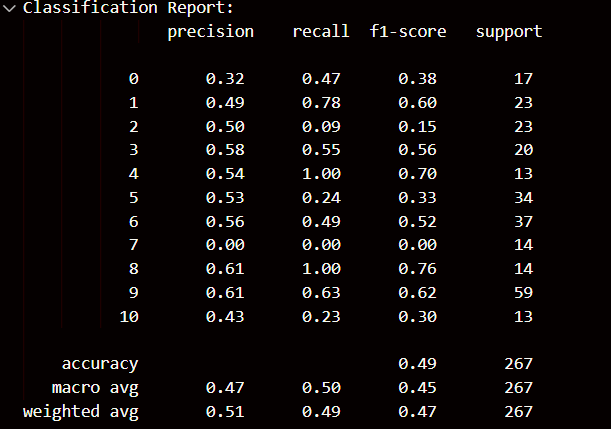
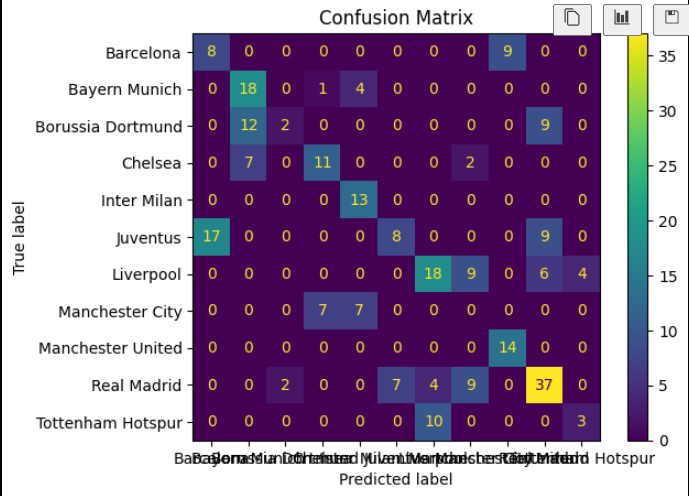
Con este algoritmo se utilizaron 4 capas ocultas dividiendo la cantidad total de los datos dedicando un 80% para el entrenamiento del modelo y el restante para probarlo, dando como resultado una precisión del 56% donde cada equipo es evaluado para ser considerado el próximo ganador, al obtener este nivel de precisión no es recomendable optar por este modelo.

** **

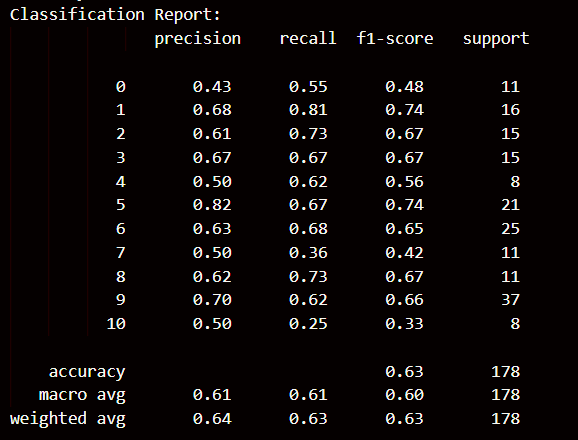
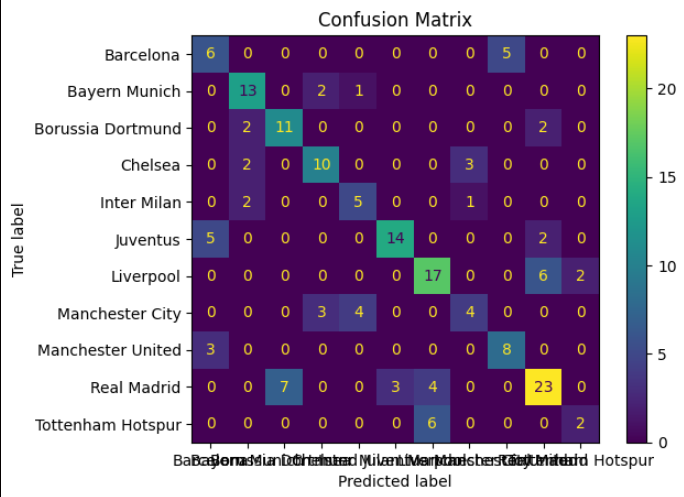
Con el siguiente algoritmo se reduce la cantidad de capas ocultas a 3, y se opta por utilizar una cantidad de neuronas mayor. Se utiliza el método de tangente para enfocarse en representaciones más abstractas y relevantes de los datos. Se mantiene la división de los datos, dedicando nuevamente el 80% de los datos para el entrenamiento del modelo y el restante para que este realice las pruebas. Se logra aumentar el nivel de precisión a un 62%, volviéndolo más eficiente a la hora de evaluar quien sería el próximo ganador. No obstante, esta cantidad no es recomendable al no ser lo suficientemente precisa para el resultado que se desea.

Se implementan más cambios como el método de activación, utilizando “relu”, mas no obstante no se consiguen los resultados esperados, siendo menos ineficiente que el método “tanh”. Este algoritmo presenta una modificación en la primera capa oculta, dándole una mayor cantidad de neuronas. También, se implementa el método “relu” y se le da una menor cantidad de datos para el entrenamiento, siendo del 70% para probar con el 30%. En este algoritmo se logra observar que tiene un menor número de iteraciones, ya que este modelo tiene una perdida en el entrenamiento y logra arrojar una precisión del 49%, lo que lo vuelve un método más ineficiente.

El siguiente modelo implementa 4 capas ocultas con la misma cantidad de neuronas y dándole un máximo de 1000 iteraciones, además de utilizar el método de activación “tanh”. Se le deja entrenar el modelo con el 80% de los datos y evaluando el restante, dando el resultado más alto con este modelo al ser de 63% de precisión.

** **

Lo que vuelve a este modelo ineficiente, es que se le da un máximo de iteraciones de 1000, sin embargo, este no logra llegar a ellas. El modelo tiene una pérdida de mejora de 0.000100, esta pérdida sucede durante 10 épocas, lo que termina haciendo que el modelo utilice solamente un máximo de 500 iteraciones o menos, lo que lo vuelve ineficiente a los resultados deseados para este modelo.

## El modelo

En función de los resultados obtenidos en el paso anterior podemos considerar al modelo “Regresión lineal” como el más apropiado para realizar nuestras predicciones.

Un método estadístico que busca predecir una variable dependiente (en este caso, el equipo ganador, winner) utilizando una o más variables independientes (características del dataset).

El objetivo principal es analizar la influencia de cada variable o combinación de variables en la predicción de los resultados.

1. Se calculan las correlaciones entre las variables del dataset para entender qué tan relacionadas están entre sí y con la variable objetivo.
   * Mapa de calor: Se utilizó un heatmap (mapa de calor) para visualizar la matriz de correlación. En este mapa:
     + Los colores más rojos indican una mayor correlación entre las variables.
     + Los colores más claros indican una menor correlación.
   * Este análisis sirve para identificar qué variables tienen mayor impacto potencial en el modelo.
2. Iteración sobre combinaciones de variables:
   * Se generaron todas las posibles combinaciones de las columnas en el dataset usando la función itertools.combinations.
   * Por cada combinación:
     + Se entrenó un modelo de regresión lineal.
     + Se calculó el coeficiente R2R^2, que mide la proporción de la varianza en la variable objetivo explicada por las variables predictoras.
   * En total, se probaron 524,287 combinaciones, lo que representa todas las posibles combinaciones de las columnas disponibles.
3. Resultados obtenidos:
   * El proceso generó combinaciones con los mejores resultados (R2R^2 más alto)

**Resultados obtenidos**

Estas combinaciones presentan los valores más altos de R2R^2, lo que significa que estas variables explican mejor el resultado del modelo:

1. **Combinación:**  
   ('Equipo', 'season', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'year', 'winner\_score', 'runner\_up\_score')  
   **Score:** 0.818 (el modelo explica el 81.8% de la varianza).
2. **Combinación:**  
   ('Año', 'Equipo', 'season', 'winner-country', 'score', 'runner-up', 'runner-up-country', 'stadium', 'final-city', 'final-country', 'attendance', 'winning-way', 'winner\_score', 'runner\_up\_score')  
   **Score:** 0.817.

Este modelo se utiliza para identificar qué combinaciones de variables permiten predecir de manera más efectiva el ganador de las finales de la UEFA Champions League. Se utilizó un enfoque de prueba exhaustiva para evaluar todas las posibles combinaciones de variables, aunque este proceso generó un gran número de resultados.

Para simplificar el análisis: Se calcularon las mejores combinaciones (variables con mayor capacidad predictiva).

## Evaluación

Este modelo en comparación del resto fue el que genera la precisión mas alta, el estándar para considerarlo como viable debe ser como mínimo un 80% y con este se cumple ese requisito. Además, esto es solo un punto de partida ya que podemos iterar todos los campos y obtener la máxima precisión.

### MAPA DE CALOR DEL MODELO



El mapa de calor muestra cómo están relacionadas las diferentes columnas del dataset entre sí usando colores:

1. **Colores del mapa:**
   * **Rojo oscuro:** Relación muy fuerte entre dos variables (correlación positiva).
   * **Blanco:** Sin relación entre las variables.
   * **Azul oscuro:** Relación inversa (cuando una variable aumenta, la otra disminuye).
2. **Diagonal principal:**
   * La diagonal principal es completamente roja porque muestra cada variable comparada consigo misma (relación perfecta, igual a 1).

Puntos importantes del mapa

1. **Relaciones fuertes (rojo oscuro):**
   * winner\_score **y** score**:** Los goles del equipo ganador están muy relacionados con el marcador total del partido.
   * runner\_up\_score **y** score**:** Los goles del subcampeón también tienen una fuerte relación con el marcador total.
   * Año **y** season**:** Estas variables son prácticamente iguales, lo que significa que se podrían eliminar redundancias.
2. **Relaciones inversas (azul oscuro):**
   * Año **y** runner\_up\_score**:** A medida que pasa el tiempo, los goles del equipo subcampeón tienden a disminuir.
   * season **y** runner\_up\_score**:** Lo mismo ocurre porque Año y season son equivalentes.
3. **Relaciones débiles (blanco o colores claros):**
   * **Variables como** Posicion**,** stadium**, y** attendance**:** Estas no tienen casi relación con otras variables, lo que indica que probablemente no son importantes para predecir el ganador.

El mapa nos ayuda a identificar:

* **Qué variables están más relacionadas con el resultado del partido (winner).**
  + Por ejemplo, score, winner\_score, y year son importantes porque tienen una relación moderada o fuerte con el ganador.
* **Qué variables no son útiles:**
  + Por ejemplo, stadium y Posición tienen poca influencia y podrían ser descartadas.

# Interpretación de los resultados

## Conclusiones

## Recomendaciones