

**Universidad  
Rey Juan Carlos**

**ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA**

**GRADO EN INGENIERÍA INFORMÁTICA**

**Curso Académico 2020/2021**

**Trabajo Fin de Grado**

**EXTRACCIÓN, INFORMATIZACIÓN Y VISUALIZACIÓN  
DE DATOS RELATIVOS A LA COMPETITIVIDAD EN  
DIFERENTES LIGAS DE FÚTBOL PROFESIONAL**

**Autor: Pablo Roldán Peigneux d'Egmont**

**Directores: Luis Rincón Córcoles**

**Alejandro J. García del Amo Jiménez**



*Dedicado a*

*Mis padres, por ser los principales responsables de que esté donde estoy ahora mismo,  
pudiendo terminar mi carrera a día de hoy.*

# Agradecimientos

Este Trabajo de Fin de Grado se lo agradezco en primer lugar a Alejandro José García del Amo Jiménez y a Luis Rincón Córcoles, mis tutores del TFG, puesto que fueron de las únicas personas que accedieron a dirigírmelo, además de mostrarme su apoyo y ayudarme durante todo su desarrollo, echándome una mano siempre que lo necesitaba, desde la elección del tema hasta la revisión final del TFG.

También he de agradecer a todos los docentes de la Universidad Rey Juan Carlos que me han dado clase, porque sin todo lo que me han aportado y enseñado, habría sido imposible desarrollar un trabajo de tal envergadura.

Por último, dar las gracias a todos los profesores de informática que he tenido en mi vida, pues supieron transmitirme la pasión que ellos sentían hacia ella, pasión que yo ahora comparro, culminando en la adquisición de un título en el Grado de Informática por la URJC, considerándome de esta manera, informático.

# Resumen

En este Trabajo Fin de Grado se presentarán las herramientas informáticas desarrolladas para realizar los cálculos presentados en el artículo titulado: “Modeling and visualizing competitiveness in soccer leagues” [1]. Este TFG añade un valor extra al TFG de Matemáticas llamado “Modelización, visualización y análisis de la competitividad en diferentes ligas de fútbol profesional” [2] realizado por mí, en el que se lleva a cabo un análisis en profundidad de dicho artículo desde el punto de vista matemático.

Para realizar dichos cálculos se implementan tres procesos: extracción de datos, informatización de cálculos y visualización de resultados.

El artículo en cuestión tratará de analizar la competitividad en diferentes ligas de fútbol, en concreto en 6 de ellas: “La Liga” (España), “Premiership” (Inglaterra), “Serie A” (Italia), “Ligue 1” (Francia), “Série A” (Brasil) y “J1 League” (Japón); durante la temporada 2018–2019. Estas serán las que se analizarán en este TFG para comprobar los resultados, además de la “Bundesliga” (Alemania) durante la temporada 2018–2019, con el fin de añadir un poco más variedad en los resultados.

Para la informatización de todo lo expuesto en el artículo, en primer lugar se realizará un proceso de extracción de datos. Para ello se hará uso de Google Sheets junto con Javascript con el fin de obtener los datos de los puntos en cada jornada en forma de tabla y llevarlos a un fichero Excel en local. Tanto para la informatización de cálculos como para la visualización de resultados, se utilizará *Python*. Este lenguaje permitirá la lectura del documento Excel, la traducción de los cálculos y la visualización de los mismos como se deseé. Una vez se haga todo esto, se compararán los resultados obtenidos con los del artículo [1].

Todo esto tiene como fin último facilitar el cálculo de la competitividad, pero en este caso, se expandirá un poco más y se compararán los resultados finales con el fin de comprobar ambos, los obtenidos en el artículo [1] y los obtenidos en este Trabajo de Fin de Grado.

# Palabras clave

## Palabras clave:

- Competitividad deportiva
- Poder complejo
- Clusterización
- Escalamiento multidimensional
- Dimensión fractal
- Entropía
- Desviación típica
- Índice de balance competitivo
- Fútbol
- Extracción de datos
- Google Sheets
- Javascript
- Excel
- Automatización de cálculos
- Visualización de datos
- Python

# Índice general

|  |           |
|--|-----------|
| <b>1. Introducción</b>                       | <b>1</b>  |
| 1.1. Motivación . . . . .                    | 2         |
| 1.2. Competencias . . . . .                  | 4         |
| <b>2. Objetivos</b>                          | <b>6</b>  |
| 2.1. Descripción del problema . . . . .      | 6         |
| 2.2. Metodología empleada . . . . .          | 7         |
| 2.2.1. Extracción de datos . . . . .         | 7         |
| 2.2.2. Informatización de cálculos . . . . . | 9         |
| 2.2.3. Visualización de resultados . . . . . | 10        |
| 2.3. Estudio de alternativas . . . . .       | 11        |
| 2.3.1. Extracción de datos . . . . .         | 11        |
| 2.3.2. Informatización de cálculos . . . . . | 12        |
| 2.3.3. Visualización de resultados . . . . . | 13        |
| <b>3. Descripción informática</b>            | <b>16</b> |
| 3.1. Especificación . . . . .                | 16        |
| 3.1.1. Extracción de datos . . . . .         | 17        |
| 3.1.2. Informatización de cálculos . . . . . | 19        |
| 3.1.3. Visualización de resultados . . . . . | 23        |
| 3.2. Diseño . . . . .                        | 26        |
| 3.2.1. Extracción de datos . . . . .         | 29        |
| 3.2.2. Informatización de cálculos . . . . . | 30        |
| 3.2.3. Visualización de resultados . . . . . | 31        |

|  |           |
|--|-----------|
| 3.3. Implementación . . . . .                | 33        |
| 3.3.1. Extracción de datos . . . . .         | 33        |
| 3.3.2. Informatización de cálculos . . . . . | 39        |
| 3.3.3. Visualización de resultados . . . . . | 45        |
| 3.4. Pruebas . . . . .                       | 48        |
| 3.4.1. Extracción de datos . . . . .         | 49        |
| 3.4.2. Visualización de resultados . . . . . | 53        |
| 3.4.3. Informatización de cálculos . . . . . | 57        |
| <b>4. Resultados obtenidos</b>               | <b>65</b> |
| <b>5. Conclusiones</b>                       | <b>75</b> |
| 5.1. Trabajos futuros . . . . .              | 85        |
| <b>Bibliografía</b>                          | <b>87</b> |

# Índice de figuras

|  |    |
|--|----|
| 2.1. Estructura de documento XML de ejemplo. . . . .   | 8  |
| 2.2. Diagrama de Shepard para “La Liga”. . . . .   | 14 |
| 2.3. Diagrama de estrés $S$ frente a la dimensión escogida $M$ para “La Liga”. . . . .   | 14 |
| 3.1. Entrada (input) y salida (output) de datos para el proceso de extracción de datos de la ligas de fútbol profesional. Tabla con los datos de “La Liga” durante la temporada 2018/2019. . . . .   | 18 |
| 3.2. Tabla con los datos de “La Liga” durante la temporada 2018/2019 en el Excel local (con conexión a Google Sheets) llamado “Conexion_DatosLigas”. . . . .   | 18 |
| 3.3. Entrada (input) y salida (output) de datos para el proceso de informatización de cálculos de manera conceptual. . . . .   | 22 |
| 3.4. Entradas de datos (inputs) para el proceso de visualización de manera conceptual.   | 26 |
| 3.5. Ejemplo de diagrama de casos de uso para un sistema (simple) de hacer consultas.  | 28 |
| 3.6. Ejemplo de diagrama de casos de uso para un sistema (complejo) en una clínica.  | 28 |
| 3.7. Diagrama de casos de uso para el sistema de extracción de datos. . . . .  | 29 |
| 3.8. Diagrama de casos de uso para el sistema de informatización de cálculos. . . . .  | 30 |
| 3.9. Diagrama de casos de uso para el sistema de visualización de resultados. . . . .  | 32 |
| 3.10. Id de todas las ligas a analizar con sus respectivos URLs. . . . .   | 34 |
| 3.11. Datos brutos obtenidos por parte del Capturador con la función “=IMPORTXML( <a href="https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/JORNADA_CORRESPONDIENTE">https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/JORNADA_CORRESPONDIENTE</a> ”, “//table[@id=’JORNADA_CORRESPONDIENTE’]//tr[2]//td[1]”)”. . . . . | 34 |
| 3.12. Equipos indexados. . . . .   | 35 |
| 3.13. Ejemplo de output del <i>ImportXML</i> . . . . .   | 36 |
| 3.14. Diferencias entre asignación, <i>copy.copy</i> y <i>copy.deepcopy</i> . . . . .  | 40 |

|   |    |
|---|----|
| 3.15. Ejemplo de cómo se obtiene un “GLCM” a partir de una matriz de grises obtenida con <i>graycomatrix</i> . . . . .  | 43 |
| 3.16. Ejemplo de imagen de los gráficos de los que se calcula la dimensión fractal. El resultado obtenido a la izquierda a partir del gráfico del poder complejo y a la derecha a partir del gráfico del MDS. Ambos de “La Liga” (España) durante la temporada 2018/2019. . . . . | 47 |
| 3.17. Ejemplo de imagen de los gráficos de los que se calcula la entropía. El resultado obtenido a la izquierda a partir del gráfico del poder complejo y a la derecha a partir del gráfico del MDS. Ambos de “La Liga” (España) durante la temporada 2018/2019. . . . .          | 47 |
| 3.18. Resultados Chievo Verona durante la temporada 2018/2019 con las funciones de “ <i>checkaPuntos</i> ” y “ <i>checkPuntosIni</i> ” sin admitir valores negativos para las puntuaciones de los equipos. . . . .  | 49 |
| 3.19. Equipo A, pierde su primer partido (1 a 2 en este ejemplo) y pasa de 0 puntos a 0 puntos. Equipo B, gana su primer partido (4 a 0 en este ejemplo) y pasa de -15 puntos a -12. . . . .  | 52 |
| 3.20. Resultados originales del MDS en todas las ligas. . . . .   | 54 |
| 3.21. Resultados del MDS tras intentar corregirlos con ejes de simetría (función “ <i>fix_MDS_ejeSim</i> ”) en todas las ligas. . . . .   | 55 |
| 3.22. Resultados con corrección del MDS con el uso de puntos intermedios (función “ <i>fix_MDS_intermedios</i> ”) de “La Liga” (España). . . . .  | 56 |
| 3.23. Resultados del <i>HICB</i> en cada una de las ligas. A la izquierda con el método correcto y a la derecha con el incorrecto. . . . .  | 57 |
| 3.24. Resultados sin corrección del MDS de la “J1 League” (Japón). . . . .  | 58 |
| 3.25. Resultados del MDS tras intentar corregirlos con ejes de simetría (función “ <i>fix_MDS_ejeSim</i> ”) de la “J1 League” (Japón). . . . .  | 60 |
| 3.26. Resultados con corrección del MDS con el uso de puntos intermedios (función “ <i>fix_MDS_intermedios</i> ”) de la “J1 League” (Japón). . . . .  | 61 |
| 3.27. Resultados sin corrección del MDS de la “Bundesliga” (Alemania). . . . .  | 62 |
| 3.28. Resultados con corrección del MDS con el uso de puntos intermedios (función “ <i>fix_MDS_intermedios</i> ”) de la “Bundesliga” (Alemania). . . . .  | 63 |

|      |   |    |
|------|---|----|
| 4.1. | Gráficos del poder complejo $S_i(k)$ de todas las ligas analizadas. . . . .   | 66 |
| 4.2. | Dendrogramas de todas las ligas analizadas. . . . .   | 67 |
| 4.3. | Clústeres resultantes a partir de los dendrogramas de todas las ligas analizadas. . . . .   | 68 |
| 4.4. | Gráficos del MDS de todas las ligas analizadas. . . . .   | 69 |
| 4.5. | Tabla con todas las medidas (columnas) de todas las ligas analizadas (filas) ordenadas de menor a mayor <i>HICB</i> . . . . .   | 70 |
| 4.6. | Gráficas de cada medida con todas las ligas analizadas. . . . .   | 72 |
| 4.7. | Matriz de correlación calculada con el coeficiente de correlación de Pearson entre todas las medidas usando los resultados de todas las ligas analizadas. . . . .   | 73 |
| 4.8. | Gráfico representativo de la matriz de correlación de las medidas de competitividad de las ligas de fútbol profesional. . . . .   | 73 |
| 5.1. | Evolución de $S_i(k)$ frente a $k$ en 6 ligas: (a) “La Liga”; (b) “Premiership”; (c) “Serie A”; (d) “Ligue 1” (e) “Série A”; (f) “J1 League”. Las lineas diagonales azules representan las ondas para valores sucesivos de $k$ . . . . .  | 76 |
| 5.2. | Dendrograma generado para “La Liga” al final de temporada ( $S_i(R)$ ) usando el agrupamiento enlace media o promedio [3] como método y la distancia euclíadiana como métrica [4]. La línea intermitente vertical de color rojo, secciona el dendrograma para obtener los diferentes clústeres, 5 en este caso. . . . . | 77 |
| 5.3. | Dendrograma generado para “La Liga” al final de temporada ( $S_i(R)$ ) usando el agrupamiento enlace media o promedio [3] como método y la distancia de Manhattan como métrica [4]. . . . .   | 78 |
| 5.4. | Dendrograma generado para “La Liga” al final de temporada ( $S_i(R)$ ) usando el agrupamiento de máximo o completo enlace [5] como método. . . . .  | 79 |
| 5.5. | Clústeres generados a partir de seccionar el dendrograma de “La Liga” en 5 grupos. El dendrograma ha utilizado el agrupamiento de máximo o completo enlace [5] como método. . . . .   | 79 |

|   |    |
|---|----|
| 5.6. Los gráficos resultantes del MDS con dimensión $M = 2$ de $S_i(k) = P_i(k) + \iota Q_i(k)$ , $0 \leq k \leq k_r$ , $k_r = 0, \dots, R$ , con la distancia de Manhattan $d_M$ para las ligas: (a) “La Liga”; (b) “Premiership”; (c) “Serie A”; (d) “Ligue 1”; (e) “Série A”; (f) “J1 League”. La tercera dimensión es calculada por RBI en función del tiempo (rondas). . . . .                   | 81 |
| 5.7. Tabla con las medidas de competitividad $b_S$ , $b_{MDS}$ , $H_S$ , $H_{MDS}$ , $\sigma$ y $HICB$ (nombradas de izquierda a derecha) para la “J1 League”, “La Liga”, “Premiership”, “Ligue 1”, “Série A” y “Serie A”, ordenadas de menor a mayor $HICB$ y $\sigma$ . . . . .   | 82 |
| 5.8. Los resultados obtenidos de $b_S$ , $b_{MDS}$ , $H_S$ , $H_{MDS}$ , $\sigma$ y $HICB$ . Las ligas están en orden descendente de competitividad según las medidas clásicas $\sigma$ y $HICB$ . Resultados basados en la tabla 5.7. . . . .  | 84 |
| 5.9. Coeficiente de correlación de Pearson $r_{xy}$ entre las medidas de competitividad $b_S$ , $b_{MDS}$ , $H_S$ , $H_{MDS}$ , $\sigma$ y $HICB$ , para la “J1 League”, “La Liga”, “Premiership”, “Ligue 1”, “Série A” y “Serie A”. Los arcos denotan los valores donde $ r_{xy}  \geq 0,9$ y los colores son proporcionales a $r_{xy}$ según el mapa de colores proporcionado a la derecha. . . . . | 84 |

# Capítulo 1

## Introducción

El fútbol es el deporte más popular del planeta con hasta 3500 millones de seguidores alrededor del mundo [6]. Un partido de fútbol profesional consta de 2 conjuntos de 11 jugadores cada uno sobre el terreno de juego, además de cambios, y uno o varios árbitros que se ocupan de que las normas se cumplan correctamente. El objetivo es hacer entrar en la portería contraria un balón que no puede ser tocado con las manos ni con los brazos, salvo por el portero en su área de meta [7]. Estos jugadores son exclusivos de ese equipo (sin contar a su selección nacional de fútbol), por lo que si un jugador  $x$  pertenece a un equipo  $A$ , este jugador no puede pertenecer a cualquier otro equipo de fútbol  $U$  (1.1). Es decir:

$$\text{jugador del equipo } A = \{x | x \in A \wedge x \notin U\}. \quad (1.1)$$

Esta regla diferencia un equipo de fútbol de una selección nacional de fútbol, puesto que muchos jugadores de fútbol, pertenecen a la vez a un equipo y a una selección nacional. Por ejemplo, Raúl González Blanco llegó a pertenecer a la vez al Real Madrid (equipo de fútbol) y a la Selección Española de Fútbol (selección nacional de fútbol) [8, 9].

Puesto que este análisis está hecho de cara a partidos de liga regular, cada partido durará un total de 2 partes de 45 minutos cada una. Si un equipo A tiene un enfrentamiento directo contra un equipo B, habrá 3 posibles resultados {ganar, empatar, perder}, donde actualmente para todas las ligas a analizar [10], y en concreto desde la temporada 95/96 para la liga española [11], cada uno de estos resultados conlleva {3, 1, 0} puntos respectivamente. Hay que anotar que la victoria del equipo A implicará la derrota del equipo B y viceversa; y si el equipo A empata, el equipo B

empatará también. Generalmente, el equipo ganador será el que haya marcado más goles que el otro, y si ambos marcan el mismo número de goles, el resultado de ambos será empate. El otro motivo por el que se puede ganar un partido es por lo que se conoce como “Walkover”, donde la victoria es otorgada a uno de los equipos si no hay más competidores, porque se retiraron, fueron descalificados, o no asistieron [12]. En el fútbol, el mínimo de jugadores de un equipo es de 7 [13], así que ya sea por falta de presentación o por expulsiones, se podría dar esta situación.

Dependiendo de estos resultados, al final de una liga regular, cuando todos los equipos se hayan enfrentado entre ellos 2 veces con cada uno, una en su estadio y otra en el del rival, se decidirá qué destino le espera a cada uno de estos equipos, que dependiendo de la liga y la temporada, será uno u otro (como veremos en el capítulo 4). En este caso, las ligas que se van a analizar son: “La Liga” (España), “Premiership” (Inglaterra), “Serie A” (Italia), “Ligue 1” (Francia), “Série A” (Brasil), “J1 League” (Japón) y “Bundesliga” (Alemania), durante la temporada 2018–2019. Es por esto, que la clasificación final de cada equipo en la tabla será lo que se tomará como el principal aspecto de interés.

Se tomarán aquellos partidos que tengan una alta incertidumbre en lo que concierne al ranking como los más competitivos, y, de la misma manera, los que menos afecten a este ranking, como los menos competitivos. Por lo que los partidos en los que ambos equipos se juegan más puestos relevantes, son aquellos en los que la competitividad es más alta [14], independientemente de rivalidades históricas, derbis, ... con el objetivo de hacer este análisis lo más objetivo posible.

## 1.1. Motivación

La motivación principal de este TFG viene de mi gran interés por el fútbol, la informática y las matemáticas. Además de que creo que todo es parametizable; por eso, al ver que se podía hacer eso mismo con algo que es a priori abstracto como la competitividad, mi interés creció instantáneamente.

No solo eso, creo que esto es de vital interés para el mercado de las apuestas deportivas. Pues partidos cuya competitividad es alta, atraerán a más consumidores. Y debido a cómo funcionan los coeficientes, cuanta más gente apueste, más probable es que obtengan beneficios, pues el margen que se llevan va acorde a la opinión de la gente [15]. Por lo que a mayor participación,

mejor se acompañará con los porcentajes estimados por la casa.

Esto se ve más claramente con un ejemplo: al tirar una moneda ideal y no trucada, uno tiene un 50 % de probabilidades de que caiga Cara y un 50 % de que caiga Cruz, y así lo opina la gente; por lo que si la casa de apuestas no quisiera llevarse nada de beneficio, entonces ofrecería una cuota de  $x_2$  en ambas apuestas (1.2). Sin embargo, si quisiese obtener un 8 % de margen de beneficio, la cuota para cada una sería de  $x_1, 85$  (1.3).

$$\text{Cuota para } 50\% \text{ sin margen de beneficio} = (0,5 \cdot 1)^{-1} = 2; \quad (1.2)$$

$$\text{Cuota para } 50\% \text{ con } 8\% \text{ de margen de beneficio} = (0,5 \cdot 1,08)^{-1} \approx 1,85. \quad (1.3)$$

Esto se hace así y no corrigiendo las cuotas al momento en función del dinero apostado a una o a otra, porque de hacerse así, estas se descompensarían si llega una persona con mucho dinero y apuesta primero a una de las dos, espera a que se corrijan, y apuesta a la contraria.

Siguiendo el ejemplo anterior y suponiendo que hubiesen 100 apostados a Cara y 100 apostados a Cruz ya, si alguien apuesta 100 a una de las 2 cuotas y luego de que se corrijan las apuestas en función del dinero, esta misma persona apuesta a la contraria otros 100, entonces garantiza su beneficio en detrimento de la casa de apuestas (ver ecuaciones 1.4, 1.5, 1.6, 1.7) como se muestra a continuación:

$$\text{Cuota para } \frac{100}{200} \text{ con } 8\% \text{ de beneficio} = \left(\frac{100}{200} \cdot 1,08\right)^{-1} \approx 1,85. \quad (1.4)$$

$$\text{Cuota para } \frac{100}{300} \text{ euros (la opuesta) con } 8\% \text{ de beneficio} = \left(\frac{100}{300} \cdot 1,08\right)^{-1} \approx 2,78. \quad (1.5)$$

$$\text{Beneficio del apostante si gana la } 1^a = 100 \cdot 1,85 - 100 = 85. \quad (1.6)$$

$$\text{Beneficio del apostante si gana la } 2^a = 100 \cdot 2,78 - 100 = 170. \quad (1.7)$$

En consecuencia, a las casas de apuestas les conviene saber en qué partidos hay mucha competitividad y en cuáles poca, puesto que en los que hay menos competitividad, debido a la poca participación, pueden correr un mayor riesgo de perder dinero, ya que, como hemos visto anteriormente, las apuestas no se corrigen en función del dinero apostado, sino en función de la opinión de la gente, así que si poca gente apuesta con mucho dinero, las apuestas corren un mayor riesgo de acabar desbalanceadas, haciéndole perder dinero a la casa de apuestas en cuestión. Un ejemplo real donde ocurrió algo similar, fue durante la temporada 2015/16, donde la inesperada victoria del Leicester City del título de la “Premiership”, provocó que la casa de apuestas “Ladbrokes”, ofreciera grandes compensaciones si los apostantes cerraban sus apuestas antes, llegando a pagar una suma superior a 3 millones de libras a los que aguantaron hasta el final [16].

Por estos dos argumentos se puede afirmar con certeza que este Trabajo Fin de Grado tiene tanto una motivación personal como empresarial, pues las casas de apuestas que se encargan de las apuestas relativas al fútbol se podrían lucrar aprovechando esta información.

## 1.2. Competencias

A continuación listaré las competencias generales y específicas que este TFG abordará [17]:

- Competencias Generales:
  - Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan por objeto la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas.
  - Capacidad para resolver problemas con iniciativa, toma de decisiones, autonomía y creatividad. Capacidad para saber comunicar y transmitir los conocimientos, habilidades y destrezas de la profesión de Ingeniero en Informática.
  - Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.

- Que los estudiantes tengan la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética.
  
- Competencias Específicas:
  - Capacidad para conocer los fundamentos teóricos de los lenguajes de programación y las técnicas de procesamiento léxico, sintáctico y semántico asociadas, y saber aplicarlas para la creación, diseño y procesamiento de lenguajes.
  - Capacidad para evaluar la complejidad computacional de un problema, conocer estrategias algorítmicas que puedan conducir a su resolución y recomendar, desarrollar e implementar aquella que garantice el mejor rendimiento de acuerdo con los requisitos establecidos.
  - Conocimiento de la estructura, organización, funcionamiento e interconexión de los sistemas informáticos, los fundamentos de su programación, y su aplicación para la resolución de problemas propios de la ingeniería.
  - Conocimiento, diseño y utilización de forma eficiente los tipos y estructuras de datos más adecuados a la resolución de un problema.

# Capítulo 2

## Objetivos

En este capítulo se detallarán los objetivos generales y específicos del Trabajo de Fin de Grado (TFG).

El objetivo principal de este TFG es el de informatizar, automatizar y visualizar los cálculos hechos en el artículo: “Modeling and visualizing competitiveness in soccer leagues” [1]. Estos cálculos tienen como fin último el de calcular la competitividad de cada liga. Todos los datos de estas ligas provienen de: [www.worldfootball.net](http://www.worldfootball.net) [18]. Una vez hechos estos cálculos, se compararán los resultados obtenidos con los obtenidos por parte del artículo en cuestión [1].

En el Trabajo Fin de Grado del grado en Matemáticas [2] asociado al presente TFG, se exploran en profundidad las herramientas usadas para este análisis de la competitividad. Sin embargo, en este TFG, el objetivo principal no es tanto el de explicar cómo son los cálculos, sino el de traducirlos a lenguaje informático, además de añadir un proceso de extracción y visualización de datos adicional.

Al ser un alumno del Doble Grado en Ingeniería Informática y Matemáticas, estas dos Memorias, relacionadas aunque diferentes, creemos que culminan de forma armoniosa los estudios realizados.

### 2.1. Descripción del problema

El problema en cuestión es el de mostrar cómo de competitivas son unas ligas en función a sus datos de puntos en cada jornada y unos cálculos ya vistos y estudiados en el artículo: “Modeling and visualizing competitiveness in soccer leagues” [1]. Para ello, se dividirá el trabajo en

3 fases: extracción de datos, informatización de cálculos y visualización de resultados.

En primer lugar, la extracción de datos pretende que de una forma sencilla, donde introduciendo no más de unos pocos datos, se pueda obtener una tabla con los equipos y sus puntos en cada jornada para una liga y una temporada concretas. Las especificaciones de lo que queremos obtener se aclararán en la parte de diseño.

En segundo lugar tenemos la fase de informatización de cálculos. En ella, basándonos en las fórmulas del artículo, trataremos de traducirlas a lenguaje informático. En concreto se usará *Python*, pues tiene un gran número de bibliotecas que nos podrán servir de apoyo para todo esto. Además, esta fase no solo trata de traducir cálculos, sino también de saber cómo procesarlos y cómo leerlos desde el método de extracción de una forma automática con una intervención mínima del usuario.

Por último, para la visualización de datos, tomaremos como referencia los gráficos del artículo, pero añadiremos sobre estos aquellos que consideremos más relevantes e interesantes también.

Estas 3 fases nos permitirán conocer cómo de bien o mal están hechos los cálculos del artículo, además de poder llevar estos cálculos a otras ligas y temporadas con facilidad.

## 2.2. Metodología empleada

A continuación se verán por encima las metodologías empleadas para cada fase, seguido de las metodologías descartadas. El motivo principal por el que se verán por encima es porque su explicación en profundidad se verá en el capítulo 3.

### 2.2.1. Extracción de datos

En primer lugar, se necesitará un método que permita extraer datos de páginas web de manera sencilla y automatizada. A ser posible, que no necesite de supervisión para evitar tener que vigilar extracciones de grandes cantidades de datos.

Los datos se obtienen de [www.worldfootball.net](http://www.worldfootball.net) [18] por ser la fuente de datos del artículo original [1], esto implica que para una misma liga en una temporada concreta, habrá que obtenerse los datos de todos los equipos en todas sus jornadas, es decir, al menos 20 equipos y 38

jornadas. Esto significa que hay que poder obtenerse los datos de al menos 38 links diferentes por liga y temporada.

La herramienta que se ha considerado más útil para todo esto es Google Sheets [19]. Los motivos principales son 4: la función *ImportXML* [20], el hecho de que las funciones en Google Sheets se ejecutan en segundo plano sin necesidad de tener el archivo abierto [21], la capacidad de poder codificar funciones con disparadores por detrás en función del tiempo usando JavaScript (esto facilitará la automatización) [22] y el hecho de tener experiencia con esta herramienta por haber trabajado ya con ella antes en proyectos personales.

*ImportXML* es una función que a partir de un *XPath*, permite obtener los datos de una página web. Un *XPath* (XML Path Language) es un lenguaje que permite construir expresiones que recorren y procesan un documento XML (ver imagen 2.1) [23]. XML es un metalenguaje que permite definir lenguajes de marcado, es decir, con etiquetas que definen su estructura [24], desarrollado por el World Wide Web Consortium (W3C) y utilizado para codificar documentos en un formato legible tanto por humanos como por máquinas [25]. Los *XPath* se obtienen a partir del HTML de una página web, en concreto al darle a inspeccionar sobre un elemento de una página web (o “Ctrl + Mayús + I”), botón derecho sobre la etiqueta de la que se quieran sus datos, “Copy > Copy *XPath*”.

Una estructura de un documento XML (imagen tomada de la referencia [25]) podría ser la siguiente:

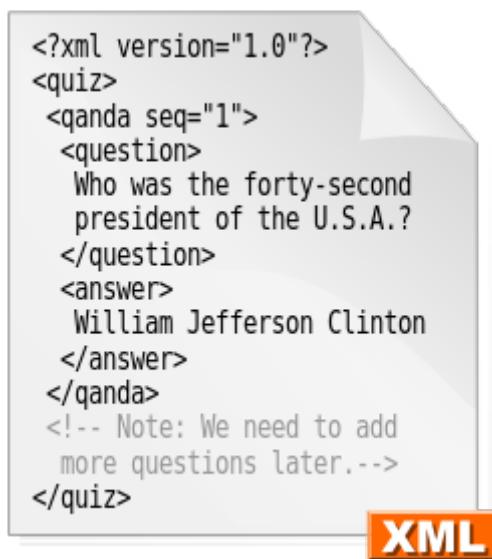


Figura 2.1: Estructura de documento XML de ejemplo.

El problema principal del *ImportXML* está en el cómo se reciben los datos, debido a que estos vienen desordenados y sin ningún tipo de formato en la mayoría de ocasiones, por ejemplo, en este caso con las ligas de fútbol tomadas desde [www.worldfootball.net](http://www.worldfootball.net) [18] (ver imagen 3.13). Esto se debe principalmente a la estructura jerárquica de los lenguajes de marcado [24], donde como lo que se quiere obtener uno son solo los valores que se enseñan por pantalla, al obtenerlos, estos vienen sin ningún tipo de separación, pues estas separaciones están en una jerarquía mucho más alta y, al intentar obtenerlas, los datos útiles se verían afectados porque se extraerían muchos más datos de los que se quieren, dificultando así un proceso de obtención de los mismos robusto y uniforme.

La forma de solucionar este problema es usando otras funciones de Google Sheets que permitan darle formato en función de cómo sean devueltos los datos a analizar.

La solución al inconveniente del formato en los datos obtenidos provenientes de [www.worldfootball.net](http://www.worldfootball.net) [18] se explicará en la sección de implementación 3.3.1.

### 2.2.2. Informatización de cálculos

Una vez obtenidos los datos, se necesitará una herramienta que permita leerlos, transformarlos y realizar los cálculos necesarios. En concreto estos cálculos serán los mencionados en el artículo: “Modeling and visualizing competitiveness in soccer leagues” [1]: clusterización, escalamiento multidimensional, entropía, dimensión fractal, desviación típica, índice de balance competitivo de Herfindahl-Hirschman y coeficiente de correlación de Pearson. Todos ellos son explicados y desarrollados en profundidad en el propio artículo [1] y en mi TFG de Matemáticas [2] asociado al presente TFG.

La herramienta por la que se ha optado es *Python* y sus diferentes librerías. Entre estas librerías encontramos: *pandas*, *numpy*, *scipy*, *sklearn* y *skimage*. A continuación desarrollaré el porqué de su uso.

La librería de *pandas* tiene muchos uso en cuanto al manejo de datos, pero aquí se utilizará principalmente por los *dataframes*. Los *dataframes* son un tipo de datos con el que se representan matrices de elementos de manera sencilla [26].

Con el mismo propósito también se utilizará *numpy*, sin embargo, para representar matrices, *numpy* tiene una estructura de *array* de 2 dimensiones en vez del de una matriz directamente. La ventaja que tiene esto es que te permite operar con *arrays* y recorrerlos puede ser más intuitivo.

Además, *numpy* funciona con la mayoría de librerías de *Python*. Pero no solo eso, sino que *numpy* cuenta con una gran cantidad de operaciones que serán útiles para la informatización de los cálculos [27]. También se usarán las librerías de *math* [28] y *cmath* [29] para otras operaciones no recogidas por *numpy*.

Siguiendo con *scipy* [30], el principal motivo del uso de esta librería se da porque gracias a las funciones de *dendrogram* [31] y *linkage* [32], se podrán obtener dendrogramas similares a los que deseamos para el cálculo de la competitividad, más concretamente, de los clústeres (grupos de elementos formados por similitud a partir de una característica en común [33]).

También se hará uso de la librería *sklearn*. La razón principal es que cuenta con funciones que permiten hacer el cálculo del escalamiento multidimensional o MDS [34]. Estas presentarán algunos problemas en sus resultados, pero serán corregidos a posteriori.

Por último, para el cálculo de las diferentes medidas de competitividad, se utilizará principalmente *numpy*, pero los datos requerirán ser leídos por la máquina y necesitarán un procesamiento a posteriori, para lo cual se hará uso de la librería *skimage* [35].

La forma en la que se utilizan estas librerías y cómo se aplican se detallará en la sección 3.3.2.

### 2.2.3. Visualización de resultados

Una vez informatizados todos los cálculos, se necesitará un método con el que visualizar los resultados. No es necesario que se haga en el mismo lenguaje de programación que los cálculos, pero hacerlos de esta forma facilitará su implementación.

Por ello se opta de nuevo por *Python*. En este caso se utilizará la librería de *pyplot* principalmente y, para un caso específico, la de *sklearn*.

La librería *pyplot* permite dibujar todo tipo de gráficas, junto con sus leyendas y ediciones para que queden como uno quiere. De hecho, incluso se pueden eliminar los ejes para dibujar figuras como veremos en uno de los casos (ver imagen 4.8) [36].

Además de esta librería, también se hará uso de *sklearn* para la clusterización, en concreto, la función *AgglomerativeClustering* [37]. De hecho, se utilizará para añadir unos gráficos no mostrados en el artículo original [1]. Estos consisten en mostrar los clústeres o grupos no solo como dendrogramas divididos por una línea vertical, sino en presentarlos como una nube de puntos, de forma que quede más claro el motivo por el que unos equipos pertenecerán a un

grupo, mientras que otros pertenecerán a uno diferente (ver imagen 4.3).

A parte de estos gráficos, también se mostrará la matriz de correlaciones, matriz que en el artículo [1] no se enseña, dejando un resultado un poco pobre al no exponer sus valores en detalle. Una matriz de correlaciones cuantifica cómo de relacionada está una variable con otra [38]. Para ello se utilizará las librería *pandas* vista en la sección 2.2.2 anterior, junto con la librería *seaborn* [39] para representar los resultados con un mapa de colores en función del valor obtenido.

En la sección 3.3.3 se explicará cómo se utilizan estas librerías en la implementación del código del proyecto. Los resultados obtenidos serán mostrados en el capítulo 4.

## 2.3. Estudio de alternativas

A la hora de realizar el presente TFG se estudiaron diversas alternativas en cuanto a la metodología que debía aplicarse a la hora de realizar la extracción de los datos, los cálculos propiamente dichos y la visualización de resultados. Todas ellas se analizarán a continuación.

### 2.3.1. Extracción de datos

Para el proceso de extracción de datos se estudiaron otras opciones como el uso de diferentes APIs para la obtención de datos web. Una API (Application Programming Interface) es un tipo de interfaz de software que ofrece un servicio (extracción de datos web en este caso) a otras piezas de software [40].

El problema principal que tenían era el hecho de ser de pago. La gran mayoría de APIs que te permiten extraer datos, por no decir todas, tienen planes de suscripción y con ellas te puedes obtener los datos que quieras o necesites, como por ejemplo [www.football-api.com](http://www.football-api.com) [41]. Además, estas tarifas suelen ser caras, pues están pensadas la mayoría para el ámbito empresarial y no el personal o estudiantil.

A su vez, es posible que de esta forma no se obtengan los datos como se desean. Por no añadir, que la experiencia con el método utilizado hacía que este quedase en el olvido, aunque el utilizado tuviera sus complicaciones también, como veremos en la sección 3.3.1.

### 2.3.2. Informatización de cálculos

En cuanto a los cálculos, se desecharon metodologías en 3 ocasiones: el cálculo de números complejos, el cálculo del MDS y la corrección de los gráficos del escalamiento multidimensional.

En lo que se refiere a los números complejos, hay un momento en el que se necesita calcular el poder complejo y para ello se necesitan los números complejos. Así que en primera instancia, eso fue lo que se hizo, calcular la tabla de puntos de este poder complejo con los números complejos tal cual viene definido en el artículo [1]. El problema de esto es que no se podía representar fácilmente en un plano y era más complicado hacer cálculos con estos puntos que de la forma por la que se optó. Aún así se mantuvo la función que calculaba de esta manera llamada “calc\_S\_i\_real”

El método elegido para el cálculo de los números complejos al final consiste en tomar la parte real como el eje de las  $X$  ( $R$ ) y la parte imaginaria como el eje de las  $Y$  ( $Q$ ). Esto además facilita los cálculos, pues se realizan sobre distancias, y es mucho más fácil calcular la distancia entre puntos de un plano, que entre números complejos. La equivalencia tomada (2.1) queda tal que:

$$\begin{aligned} Victoria &: 3 + i0 \rightarrow (3, 0); \\ Empate &: 1 + i2 \rightarrow (1, 2); \\ Derrota &: 0 + i3 \rightarrow (0, 3), \end{aligned} \tag{2.1}$$

donde  $i = \sqrt{-1}$ .

En el cálculo del MDS se utiliza una función que tiene un componente aleatorio como es `sklearn.manifold.MDS` [42]. Antes de observar este hecho, se observó que se obtenían resultados diferentes al calcular los gráficos del MDS con los valores de la matriz transpuesta en vez de con los de la original. Una matriz se transpone al intercambiar sus filas por sus columnas ( $A : 3x2 \rightarrow A \text{ transpuesta} : 2x3$ ) [43].

Una vez predeterminado el factor aleatorio para que fuera siempre el mismo, se observó que daba igual qué método se utilizase, porque aunque se obtenían resultados diferentes con la matriz original y la transpuesta, ambos se referían a lo mismo. Por ello al final se optó por el uso de la matriz original, ya que es más intuitivo hacer los cálculos directamente, frente a

transponer la matriz, hacer los cálculos y volver a transponer la matriz para que se quede como debe. De todos modos e igual que en el caso anterior, se mantuvo la función que utiliza la matriz transpuesta, esta se llama “`calc_MDS_T`”

Por último, estos resultados daban errores o resultados que estaban en la parte opuesta del gráfico porque la función `sklearn.manifold.MDS` así lo estimaba. Para corregir estos errores, en primer lugar se optó por sacar el punto perpendicular a un eje de simetría obtenido a partir del resto de puntos. Esta idea, aunque mejoraba un poco el resultado, no era buena. Al igual que en los casos anteriores, se puede encontrar esta función como “`fix_MDS_ejeSim`”.

Así que al final se optó porque si había puntos en el lado opuesto, saltárselos y en vez de usar los puntos mal colocados, usar los puntos intermedios que se encontraban entre los que estaban bien colocados. Esto se hace con la función “`fix_MDS_intermedios`”.

Todas las funciones aquí mencionadas serán explicadas y desarrolladas en profundidad en la sección 3.3.2 para clarificar los métodos utilizados.

### 2.3.3. Visualización de resultados

Por último, en lo que respecta a la visualización de los gráficos, también se descartan algunos gráficos que si que muestra el artículo original [1]. En concreto 2 de ellos: el diagrama de Shepard y la gráfica que enfrenta el estrés ( $S$ ) contra el número de dimensiones ( $M$ ).

El diagrama de Shepard (ver figura 2.2) es un gráfico que nos permite conocer cómo de buenos o malos son nuestros resultados. La cuestión es que ya sabemos que este método es bueno por lo que describe el artículo [1] y lo vamos a utilizar de todos modos, así que lo que nos interesa es mostrar los resultados y no una gráfica que analice su validez. La gráfica en cuestión es la siguiente (figura 2.2 tomada de la sección 3 de [1]):

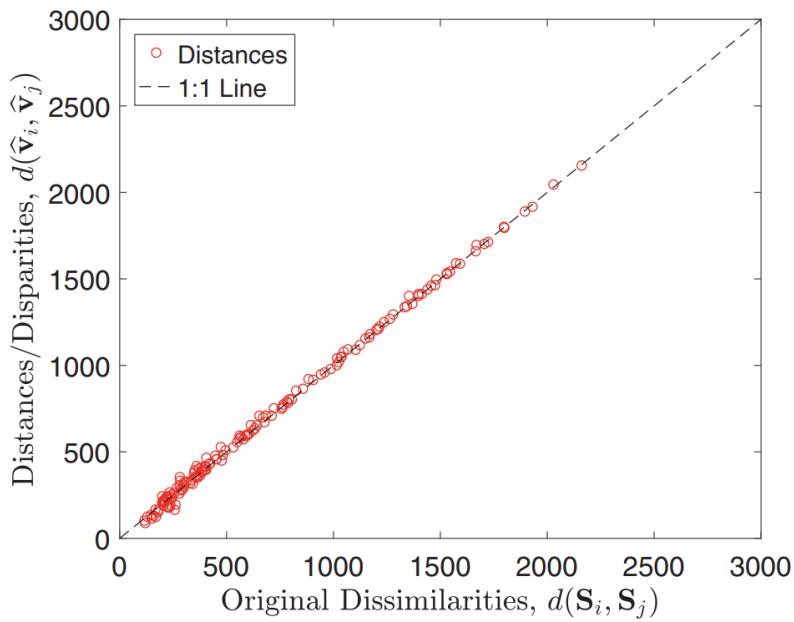


Figura 2.2: Diagrama de Shepard para “La Liga”.

Del mismo modo tenemos la gráfica de  $S$  frente a  $M$  (ver figura 2.3), que analiza si merece la pena mejorar los resultados a cambio de complicar los gráficos resultantes. Así que, puesto que de nuevo analiza la validez de los resultados en vez de mostrarlos, la gráfica de  $S$  frente a  $M$  también es descartada. La gráfica en cuestión es la siguiente (figura 2.3 tomada de la sección 3 de [1]):

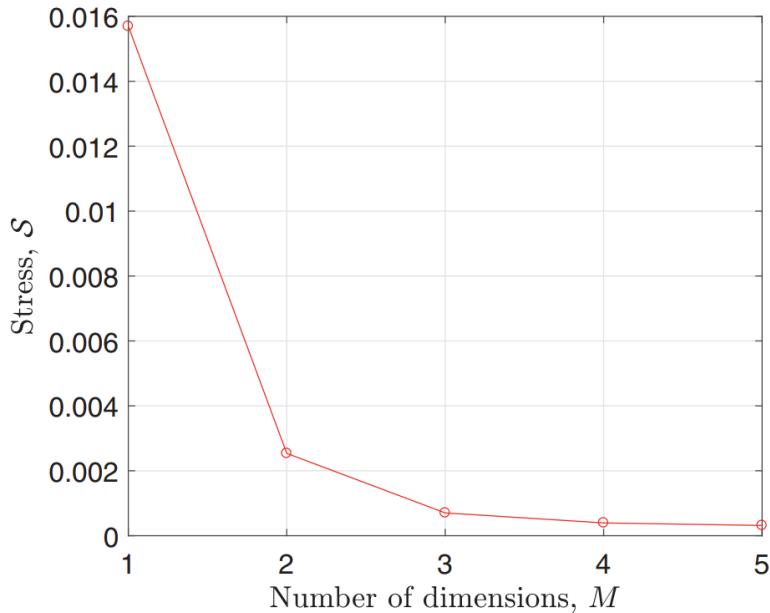


Figura 2.3: Diagrama de estrés  $S$  frente a la dimensión escogida  $M$  para “La Liga”.

Se puede concluir que como lo que en este TFG se busca es la visualización de los resultados y no la comprobación de si estos son buenos o malos (eso es cosa del artículo [1] y del TFG de Matemáticas [2]), se opta por descartar estos 2 gráficos directamente.

Cabe añadir que del mismo modo que se descartan gráficos del artículo [1], también se añaden nuevos no utilizados en este mismo artículo como se menciona en la sección 2.2.3.

# Capítulo 3

## Descripción informática

En esta sección se profundizará y se detallarán las metodologías mencionadas en el capítulo de Objetivos. Asimismo, se seguirá una estructura de 4 ciclos ordenada para cada una de las 3 fases. Esta estructura consta de: especificación, diseño, implementación y pruebas. Todas ellas explicadas en profundidad a continuación

Como se menciona en el capítulo de Objetivos, la idea es informatizar todos los métodos y cálculos del artículo: “Modeling and visualizing competitiveness in soccer leagues” [1]. Es por esto que en este capítulo, el objetivo no es el de explicar y desarrollar dicho artículo, sino exponer de manera clara cómo se ha informatizado lo que en él se muestra.

### 3.1. Especificación

La especificación consiste principalmente en conocer los requisitos del usuario. Para esto hay que conocer en primer lugar quiénes son estos usuarios. En este caso, puesto que el es el desarrollador el único usuario al que está destinado este programa, no será necesario ningún método de recogida de requisitos en masa. Esto no exime de que se recojan requisitos, pero estos serán mucho más simples de obtenerse y podrán acomodarse en función de la implementación. El método de recogida de requisitos ha consistido en conocer y anotar qué datos de entrada y de salida se requieren, y en función de estos, hacer el diseño, la implementación y las pruebas [44].

### 3.1.1. Extracción de datos

En primer lugar, para la extracción de datos necesitamos saber qué va a requerir nuestro sistema. Tras un análisis de los gráficos y los cálculos, se puede observar que se requiere obtener el progreso de cada equipo en cada momento de la liga, es decir, los puntos de cada equipo desde la jornada 0 a la última.

Para ello, lo ideal es crear un sistema que permita extraer estos datos introduciendo únicamente el nombre de la liga y la temporada en cuestión. Este tendrá que obtenerse los datos de todos los equipos en todas las jornadas de esa liga en concreto, por lo que se optará por una tabla con equipos en un eje y las jornadas en el otro. También será conveniente que el sistema en cuestión funcione en segundo plano sin necesidad de vigilancia alguna.

Una vez hecho esto, puesto que *Python* no podrá leer de una página web, se necesita un método que permita trasladar estos resultados a un archivo local. Para ello se conectará el resultado obtenido en Google Sheets con un archivo Excel local y se establecerá que se actualice de forma automática.

Por último, se permite que todo este sistema no sea muy amigable con el usuario (not user friendly), pues como se ha mencionado anteriormente, el usuario es el mismo que el desarrollador, así que no hace falta que tenga una aplicación por detrás que facilite su uso, vale con que se pueda introducir la liga y la temporada de alguna forma, y, con ello, obtener el resto de datos. El método empleado será introducir una URL como se ve en la imagen 3.1.

Recapitulando, el sistema de extracción de datos ha de tener lo siguiente:

- Requisitos de entrada (Input requirements):
  - Liga en cuestión (no necesariamente user friendly)
  - Temporada de dicha liga (no necesariamente user friendly)
  - Extracción de datos no supervisada
- Requisitos de salida (Output requirements):
  - Tabla con los datos de una liga en una temporada concreta de dimensiones:  
*Jornadas × Equipos*
  - Resultados en local

Por todo esto, lo que se desea tendrá que ser tal que así (imagen tomada del proyecto desarrollado final en Google Sheets llamado “Rastrealigas” [45]):



**Input:**

| # (1-7) | ID_Liga                         | URL   | Es Algo (EN USO) | Nombre_Liga | Temporada | País   |
|---------|---------------------------------|---|------------------|-------------|-----------|--------|
| 1       | esp-primer-a-division-2018-2019 | <a href="https://www.worldfootball.net/schedule/esp-primer-a-division-2018-2019-spielzeit/">https://www.worldfootball.net/schedule/esp-primer-a-division-2018-2019-spielzeit/</a> |                  | 1 La Liga   | 2018-2019 | España |

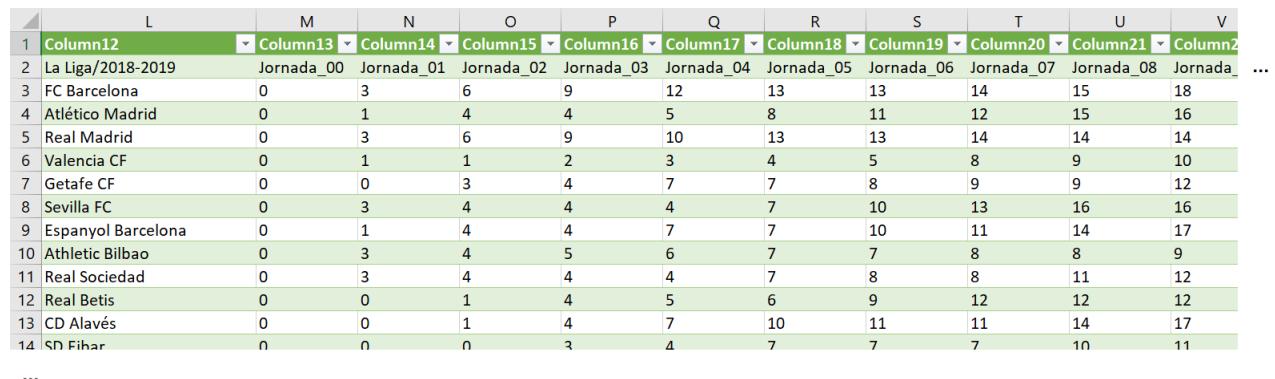
**Output:**

| La Liga/2018-2019 | Jornada_00 | Jornada_01 | Jornada_02 | Jornada_03 | Jornada_04 | Jornada_05 | Jornada_06 | Jornada_07 | Jornada_08 | Jornada_09 | Jornada_10 | Jornada_... |
|-------------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|-------------|
| FC Barcelona      | 0          | 3          | 6          | 9          | 12         | 13         | 13         | 14         | 15         | 18         | 21         | 24          |
| Atlético Madrid   | 0          | 1          | 4          | 4          | 5          | 8          | 11         | 12         | 15         | 16         | 19         | 20          |
| Real Madrid       | 0          | 3          | 6          | 9          | 10         | 13         | 13         | 14         | 14         | 14         | 17         |             |
| Valencia CF       | 0          | 1          | 1          | 2          | 3          | 4          | 5          | 8          | 9          | 10         | 11         | 11          |
| Getafe CF         | 0          | 0          | 3          | 4          | 7          | 7          | 8          | 9          | 9          | 12         | 15         | 16          |
| Sevilla FC        | 0          | 3          | 4          | 4          | 4          | 7          | 10         | 13         | 16         | 16         | 19         | 20          |
| Espanyol Barce    | 0          | 1          | 4          | 4          | 7          | 7          | 10         | 11         | 14         | 17         | 18         | 21          |
| Athletic Bilbao   | 0          | 3          | 4          | 5          | 6          | 7          | 7          | 8          | 8          | 9          | 10         |             |
| Real Sociedad     | 0          | 3          | 4          | 4          | 4          | 7          | 8          | 8          | 11         | 12         | 12         | 13          |
| Real Betis        | 0          | 0          | 1          | 4          | 5          | 6          | 9          | 12         | 12         | 12         | 13         |             |
| CD Alavés         | 0          | 0          | 1          | 4          | 7          | 10         | 11         | 11         | 14         | 17         | 20         | 20          |
| SD Eibar          | 0          | 0          | 0          | 3          | 4          | 7          | 7          | 7          | 10         | 11         | 11         | 14          |
| ...               | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...         |

Figura 3.1: Entrada (input) y salida (output) de datos para el proceso de extracción de datos de la ligas de fútbol profesional. Tabla con los datos de “La Liga” durante la temporada 2018/2019.

Lo único que se necesita como input realmente es lo que está en el cuadro de URL marcado en rojo. El resto de datos no son necesarios, pero serán de gran utilidad para dejar el resultado final con más detalles. En los resultados se ven los equipos pertenecientes a “La Liga” (España) en las filas de color naranja (20 en total en este caso) y en las columnas, en amarillo, las jornadas (38 más la jornada 0, en total 39 en este caso).

Por último, tras hacer la conexión de Google Sheets con un fichero Excel local, se necesita principalmente que se mantenga la estructura anterior, siendo el resultado esperado y obtenido el siguiente (imagen 3.2 tomada del Excel local llamado “Conexion\_DatosLigas”):



| 1   | L                  | M          | N          | O          | P          | Q          | R          | S          | T          | U          | V           |
|-----|--------------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|-------------|
| 2   | Column12           | Column13   | Column14   | Column15   | Column16   | Column17   | Column18   | Column19   | Column20   | Column21   | Column2     |
| 3   | La Liga/2018-2019  | Jornada_00 | Jornada_01 | Jornada_02 | Jornada_03 | Jornada_04 | Jornada_05 | Jornada_06 | Jornada_07 | Jornada_08 | Jornada_... |
| 4   | FC Barcelona       | 0          | 3          | 6          | 9          | 12         | 13         | 13         | 14         | 15         | 18          |
| 5   | Atlético Madrid    | 0          | 1          | 4          | 4          | 5          | 8          | 11         | 12         | 15         | 16          |
| 6   | Real Madrid        | 0          | 3          | 6          | 9          | 10         | 13         | 13         | 14         | 14         |             |
| 7   | Valencia CF        | 0          | 1          | 1          | 2          | 3          | 4          | 5          | 8          | 9          | 10          |
| 8   | Getafe CF          | 0          | 0          | 3          | 4          | 7          | 7          | 8          | 9          | 9          | 12          |
| 9   | Sevilla FC         | 0          | 3          | 4          | 4          | 4          | 7          | 10         | 13         | 16         | 16          |
| 10  | Espanyol Barcelona | 0          | 1          | 4          | 4          | 7          | 7          | 10         | 11         | 14         | 17          |
| 11  | Athletic Bilbao    | 0          | 3          | 4          | 5          | 6          | 7          | 7          | 8          | 8          | 9           |
| 12  | Real Sociedad      | 0          | 3          | 4          | 4          | 4          | 7          | 8          | 8          | 11         | 12          |
| 13  | Real Betis         | 0          | 0          | 1          | 4          | 5          | 6          | 9          | 12         | 12         | 12          |
| 14  | CD Alavés          | 0          | 0          | 1          | 4          | 7          | 10         | 11         | 11         | 14         | 17          |
| 15  | SD Eibar           | 0          | 0          | 0          | 3          | 4          | 7          | 7          | 7          | 10         | 11          |
| ... | ...                | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...        | ...         |

Figura 3.2: Tabla con los datos de “La Liga” durante la temporada 2018/2019 en el Excel local (con conexión a Google Sheets) llamado “Conexion\_DatosLigas”.

En dicha imagen se pierden los colores pero se muestra exactamente lo mismo que en la imagen 3.1, pérdida en los detalles que es irrelevante a la hora de leer los datos para un lenguaje de programación como *Python*. Lo único relevante es que los datos estén en local, objetivo que se consigue con esta solución.

Aquí se han mostrado los resultados finales con el fin de clarificar qué se deseaba como salida de datos (output), pero en la práctica hay que tener todas estas ideas claras antes de seguir con el siguiente ciclo, el ciclo de diseño 3.2.

### 3.1.2. Informatización de cálculos

Una vez obtenidos los datos en forma de tabla, se necesita un programa o software que sea capaz de leerlos y procesarlos para los cálculos de después.

Para la lectura se opta por una lista de tablas o matrices como la que se puede observar en la imagen 3.3. A partir de estas tablas de datos, es necesario poder obtener una similar pero que represente el poder complejo (explicado en el artículo original [1] y en el TFG de Matemáticas [2], los resultados con una breve explicación están en el capítulo 4). Esta tabla ha de tener las mismas dimensiones, pero con puntos en un plano ( $x, y$ ) en cada una de sus casillas en vez de valores sueltos. Estos puntos representarán los mismo que los valores, pues se obtienen a partir de ellos.

No solo hará falta una matriz de puntos, sino su respectiva matriz de distancias. Una matriz de distancias es una matriz simétrica que representa la distancia de un punto con cada uno de los otros, por lo que tiene 0's en su diagonal principal [46]. Esta será necesaria para poder obtenerse el gráfico resultante del proceso de escalamiento multidimensional o MDS. De esta manera, como lo que se quiere obtener es el gráfico resultante del MDS, se necesitan conocer también las coordenadas del mismo.

Una vez hecho esto, se tienen que calcular la entropía ( $H$ ) y la dimensión fractal ( $b$ ) de cada uno de los gráficos obtenidos. Hay 2 gráficos por liga, uno del poder complejo  $S_i$  y otro del MDS, por lo que se han de obtener 4 medidas en total: la dimensión fractal del gráfico del poder complejo ( $b_{S_i}$ ), la dimensión fractal del gráfico del MDS ( $b_{MDS}$ ), la entropía del gráfico del poder complejo ( $H_{S_i}$ ) y la entropía del gráfico del MDS ( $H_{MDS}$ ). Estas medidas se compararán con las 2 medidas clásicas de competitividad: la desviación típica ( $\sigma$ ) y el índice de balance competitivo de Herfindahl-Hirschman ( $HICB$ ). Por todo ello se requerirá un tabla

como resultado final, donde en un eje se encuentren las 7 ligas analizadas, mientras que en el otro estén estas 6 medidas. No se entra en detalle sobre qué significan exactamente cada una de estas medidas, dado que ya están explicadas en el TFG de Matemáticas complementario [2], y únicamente se describen por encima en el capítulo 4.

Por último, una vez obtenida esta tabla, hay que calcularse el coeficiente de correlación de Pearson ( $r_{xy}$ ). Este coeficiente es un índice que se utiliza con el fin de medir el grado de relación entre 2 variables cuantitativas y continuas [47], es decir, cómo de relacionadas están unas variables con las otras y viceversa. El resultado esperado es una matriz de correlación simétrica que tenga en ambos ejes las 6 medidas y en la diagonal principal todo 1's, pues una medida está completamente correlada consigo misma [38].

Recapitulando, el sistema de informatización de cálculos ha de tener lo siguiente:

- Requisitos de entrada (Input requirements):
  - Tabla (en local) con los datos de una liga en una temporada concreta de dimensiones:  
 $Jornadas \times Equipos$
- Requisitos de salida (Output requirements):
  - Lista de Ligas de matrices con:  $Jornadas \times Equipos$
  - Tabla de coordenadas del gráfico del poder complejo ( $S_i$ )
  - Matriz de distancias de las coordenadas del poder complejo ( $S_i$ )
  - Tabla de coordenadas del gráfico del escalamiento multidimensional ( $MDS$ )
  - Tabla con las medidas de competitividad de cada liga:  $Medidas \times Ligas$
  - Matriz de correlación entre las medidas de competitividad

De forma más visual, las especificaciones de la informatización de los cálculos buscan obtenerse algo así (imagen 3.3):

**Input:**

| L                    | M          | N          | O          | P          | Q          | R          | S          | T          | U          | V           |
|----------------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|-------------|
| Column12             | Column13   | Column14   | Column15   | Column16   | Column17   | Column18   | Column19   | Column20   | Column21   | Column22    |
| 2 La Liga/2018-2019  | Jornada_00 | Jornada_01 | Jornada_02 | Jornada_03 | Jornada_04 | Jornada_05 | Jornada_06 | Jornada_07 | Jornada_08 | Jornada_... |
| 3 FC Barcelona       | 0          | 3          | 6          | 9          | 12         | 13         | 14         | 15         | 16         | 18          |
| 4 Atlético Madrid    | 0          | 1          | 4          | 4          | 5          | 8          | 11         | 12         | 15         | 16          |
| 5 Real Madrid        | 0          | 3          | 6          | 9          | 10         | 13         | 13         | 14         | 14         | 14          |
| 6 Valencia CF        | 0          | 1          | 1          | 2          | 3          | 4          | 5          | 8          | 9          | 10          |
| 7 Getafe CF          | 0          | 0          | 3          | 4          | 7          | 7          | 8          | 9          | 9          | 12          |
| 8 Sevilla FC         | 0          | 3          | 4          | 4          | 4          | 7          | 10         | 13         | 16         | 16          |
| 9 Espanyol Barcelona | 0          | 1          | 4          | 4          | 7          | 7          | 10         | 11         | 14         | 17          |
| 10 Athletic Bilbao   | 0          | 3          | 4          | 5          | 6          | 7          | 7          | 8          | 8          | 9           |
| 11 Real Sociedad     | 0          | 3          | 4          | 4          | 4          | 7          | 8          | 8          | 11         | 12          |
| 12 Real Betis        | 0          | 0          | 1          | 4          | 5          | 6          | 9          | 12         | 12         | 12          |
| 13 CD Alavés         | 0          | 0          | 1          | 4          | 7          | 10         | 11         | 11         | 14         | 17          |
| 14 Villarreal        | 0          | 0          | n          | n          | 4          | 4          | 7          | 7          | 7          | 11          |

...



|        |        |        |        |        |        |        |
|--------|--------|--------|--------|--------|--------|--------|
| Liga 1 | Liga 2 | Liga 3 | Liga 4 | Liga 5 | Liga 6 | Liga 7 |
|--------|--------|--------|--------|--------|--------|--------|



|         |          |
|---------|----------|
| Liga 1  | Jornadas |
| Equipos |          |

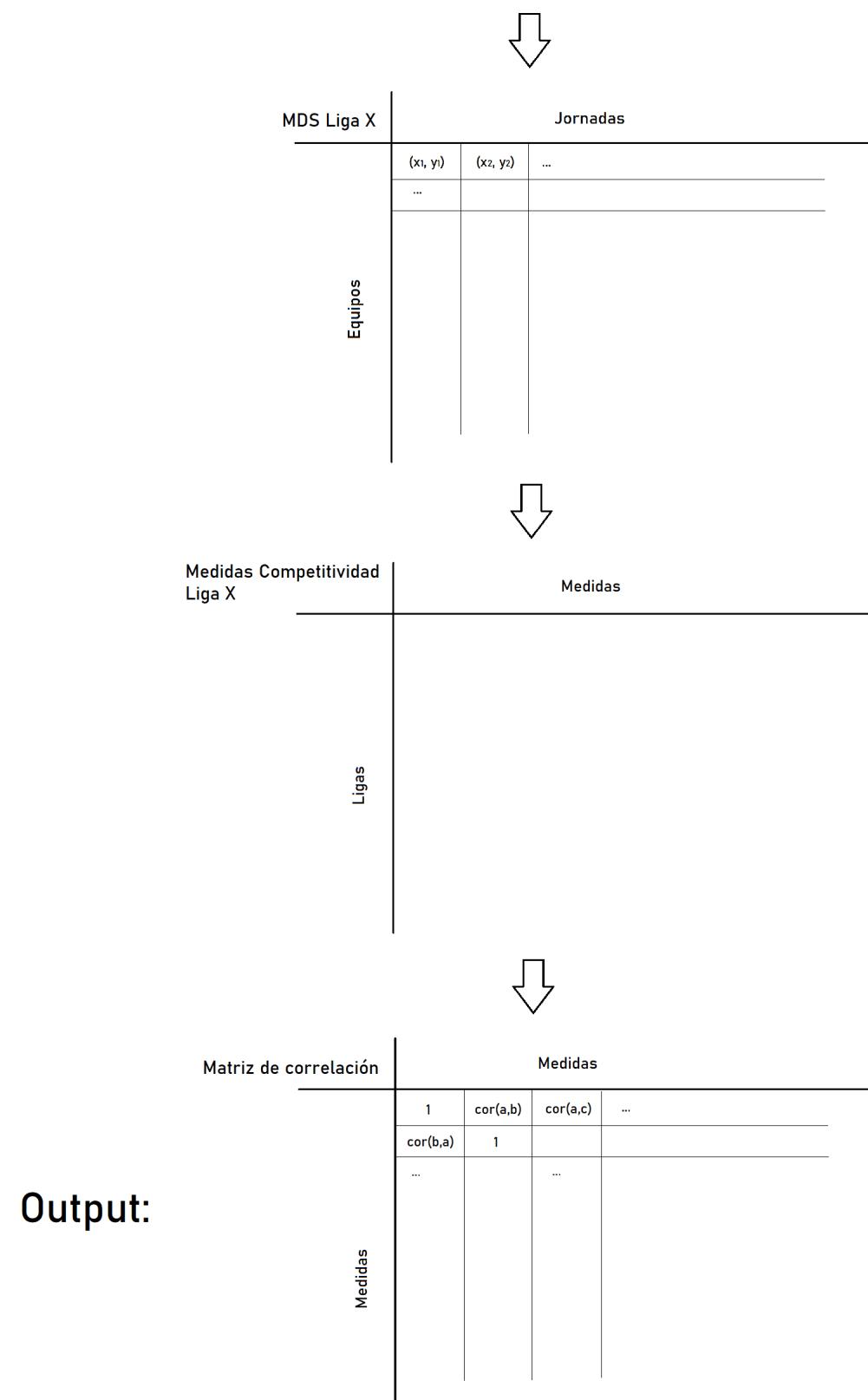
|         |          |
|---------|----------|
| Liga 7  | Jornadas |
| Equipos |          |



| Poder Complejo Liga X |                                    | Jornadas                           |     |  |
|-----------------------|------------------------------------|------------------------------------|-----|--|
| Equipos               | (x <sub>1</sub> , y <sub>1</sub> ) | (x <sub>2</sub> , y <sub>2</sub> ) | ... |  |
|                       | ...                                |                                    |     |  |
|                       |                                    |                                    |     |  |



| Matriz de distancias<br>(Si) Liga X |        | Coordenadas |        |     |
|-------------------------------------|--------|-------------|--------|-----|
| Coordenadas                         | 0      | d(a,b)      | d(a,c) | ... |
|                                     | d(b,a) | 0           |        |     |
|                                     | ...    |             | ...    |     |
|                                     |        |             |        |     |



**Output:**

Figura 3.3: Entrada (input) y salida (output) de datos para el proceso de informatización de cálculos de manera conceptual.

Con los datos de todas estas diferentes tablas se podrán hacer gráficos que permitan visualizar los resultados. Todos los resultados obtenidos a partir de ellas se encuentran en el capítulo 4.

### 3.1.3. Visualización de resultados

En último lugar, una vez se obtienen estas tablas, hay que decidir sus representaciones. Muchas de ellas serán muy intuitivas, pues son tablas de coordenadas. Por lo general se imitarán muchas de las representaciones del artículo original [1].

Empezando con la tabla de coordenadas del poder complejo, esta se representará con un gráfico con todos los puntos de la tabla, pero para que se entienda el progreso de cada equipo, en vez de trazar líneas por jornadas, estas se trazarán según los equipos, es decir, una línea por cada equipo.

Con esta misma tabla de coordenadas habrá que obtenerse el dendrograma o esquema en forma de árbol con el que se culminará el proceso de clusterización o agrupamiento. Además, para clarificar este dendrograma, se añadirá un gráfico no mostrado en el artículo original [1]. Este consiste en colorear los puntos pertenecientes a cada grupo o clúster de diferentes colores, creando así nubes de puntos que clarificará por qué cada equipo estará en cada grupo de una forma más intuitiva.

Para tabla de coordenadas resultante del proceso de escalamiento multidimensional se trazarán los puntos y las líneas de manera idéntica a la tabla de coordenadas del poder complejo. Esto quiere decir que se dibujarán todos los puntos y se unirán aquellos que pertenezcan al mismo equipo. El gráfico resultante tendrá eje  $X$  e  $Y$ , pero ambos son irrelevantes, pues lo que importa realmente es la distancia que tiene cada punto con el resto [48].

Con la tabla de medidas se querrá obtener un gráfico para cada medida, por lo que se representarán 6 gráficos en total. En el eje  $X$  se colocarán las ligas de fútbol profesional que son variables discretas, es decir, no pueden tomar ningún valor entre dos consecutivos [49]; y en el eje  $Y$  se pondrán los valores de la medida que muestre el gráfico en cuestión.

En último lugar se querrá representar la matriz de correlación de las medidas de competitividad con un gráfico. Para ello se colocarán las 6 medidas en forma de hexágono y se dibujarán líneas de colores diferentes en función de la correlación que tengan las variables entre sí.

Además del gráfico, un mapa de calor de esta matriz será de gran utilidad a la hora de visua-

lizar los valores con más detalles. Este, igual que el gráfico anterior, tendrá colores diferentes según la correlación entre dos variables y estos deberán ser análogos a los del gráfico anterior.

Recapitulando, el sistema de visualización de resultados necesitará lo siguiente:

- Requisitos de entrada (Input requirements):
  - Tabla de coordenadas del gráfico del poder complejo ( $S_i$ )
  - Tabla de coordenadas del gráfico del escalamiento multidimensional ( $MDS$ )
  - Tabla con las medidas de competitividad de cada liga: *Medidas × Ligas*
  - Matriz de correlación entre las medidas de competitividad
- Requisitos de salida (Output requirements):
  - Gráfico del poder complejo ( $S_i$ ) unido por equipos
  - Dendrograma resultante de la clusterización
  - Visualización de los clústeres formados en el gráfico del poder complejo ( $S_i$ ) con nubes de puntos
  - Gráfico del escalamiento multidimensional ( $MDS$ ) unido por equipos
  - Gráficos de cada medida de competitividad (6 medidas) con cada liga
  - Gráfico de líneas entre las medidas de competitividad según la matriz de correlación
  - Mapa de calor entre las medidas de competitividad según la matriz de correlación

Las entradas de datos (inputs) serán las mismas que las vistas en la sección 3.1.2, pero no se utilizarán todas ellas. En concreto las especificaciones de la visualización de resultados tendrán lo siguiente (imagen 3.4):

**Input 1:**

| Equipos | Jornadas                           |                                    |     |
|---------|------------------------------------|------------------------------------|-----|
|         | (x <sub>1</sub> , y <sub>1</sub> ) | (x <sub>2</sub> , y <sub>2</sub> ) | ... |
|         | ...                                |                                    |     |
|         |                                    |                                    |     |

**Input 2:**

| Equipos | Jornadas                           |                                    |     |
|---------|------------------------------------|------------------------------------|-----|
|         | (x <sub>1</sub> , y <sub>1</sub> ) | (x <sub>2</sub> , y <sub>2</sub> ) | ... |
|         | ...                                |                                    |     |
|         |                                    |                                    |     |

**Input 3:**

| Ligas | Medidas                |        |         |
|-------|------------------------|--------|---------|
|       | Medidas Competitividad | Liga X | Medidas |
|       | ...                    |        |         |
|       |                        |        |         |

**Input 4:**

| Matriz de correlación |          | Medidas |          |          |     |
|-----------------------|----------|---------|----------|----------|-----|
|                       |          | 1       | cor(a,b) | cor(a,c) | ... |
| Medidas               | cor(b,a) | 1       |          |          |     |
|                       | ...      |         | ...      |          |     |
|                       |          |         |          |          |     |

Figura 3.4: Entradas de datos (inputs) para el proceso de visualización de manera conceptual.

La salida de datos (output) resultante o esperada de la especificación no se incluye aquí, pues son los resultados obtenidos al final de todo el proyecto, por lo que se verán en el capítulo 4.

## 3.2. Diseño

Siguiendo con el diseño, este trata principalmente de hacerse una idea abstracta de cómo será la implementación en función de las especificaciones. Para ello, serán de gran utilidad los distintos diagramas UML (Unified Modeling Language) utilizados en la ingeniería de software [50]. Es decir, si la especificación trata de describir el qué hay que hacer, el diseño describe el cómo [44].

En concreto se utilizará el diagrama UML de casos de uso [51]. Este tipo de diagrama en concreto constará de 4 elementos diferentes: actores, casos de uso, relaciones y límites del sistema [52].

En primer lugar están los actores, quienes se definen como toda entidad externa al sistema que guarda una relación con este y que le demanda una funcionalidad. Esto incluye a los operadores humanos pero también incluye a todos los sistemas externos, además de entidades abstractas, como el tiempo. En el caso de los seres humanos se pueden ver a los actores como definiciones de rol por lo que un mismo individuo puede corresponder a uno o más actores [53]

[51].

Siguiendo con los casos de uso, quienes dan nombre a este tipo de diagramas, harán referencia a las acciones ejecutables en el sistema y se representarán con un círculo. Estas responden a las funcionalidades del sistema y son realizadas por otros actores o por otras funcionalidades [54]. Quién ejecuta las Acciones vendrá definido por las relaciones.

Las relaciones indican quién ejecuta los casos de uso además de poder aportar información adicional. Se representan con líneas rectas entre los casos de uso y los actores en todas las combinaciones posibles entre ellos: casos de uso-casos de uso, casos de uso-actores y actores-actores. Hay 4 tipos diferentes de relaciones según su funcionalidad: comunica, << *include* >>, << *extend* >>, y generalización [55].

Las relaciones que comunican marcan la asociación entre un actor y un caso de uso que denota la participación del actor en dicho caso de uso (entre actor y acción); los << *include* >> definen una relación de dependencia entre dos casos de uso que implica la inclusión del comportamiento de un escenario en otro (entre acción y acción); los << *extend* >> describen una subfunción de los casos de uso, normalmente un nuevo comportamiento no previsto en un caso de uso existente (entre actor y acción); y, para finalizar, los de generalización indican que un caso de uso es una variante de otro [56] [55].

De forma más clara, los << *include* >> marcan una sucesión de acciones que dependen cada una de la anterior como podrían ser: colocar la tarjeta en el cajero automático, ingresar el PIN y que aparezca el menú principal; mientras que los << *extend* >> aportan una funcionalidad extra y no esperada por el caso de uso que extienden, como cancelar una operación de ingreso o extracción de dinero en el ejemplo anterior del cajero automático [57].

En último lugar estarán los límites del sistema, estos se encargarán de delimitar qué casos de uso pertenecen o no al sistema representado. Para ello se usará un rectángulo que contendrá a los casos de uso y dejará fuera a los actores, siendo las relaciones el único elemento que lo puede cruzar.

Un ejemplo simple de diagrama de casos de uso será el siguiente (imagen 3.5 tomada de [55]):



Figura 3.5: Ejemplo de diagrama de casos de uso para un sistema (simple) de hacer consultas.

En el ejemplo del diagrama no hay límites del sistema, pero de haberlos, estos delimitarían los casos de uso dejando fuera a los actores con un rectángulo.

Y un ejemplo más completo de este tipo de diagramas será el siguiente (imagen 3.6 tomada de [52]):

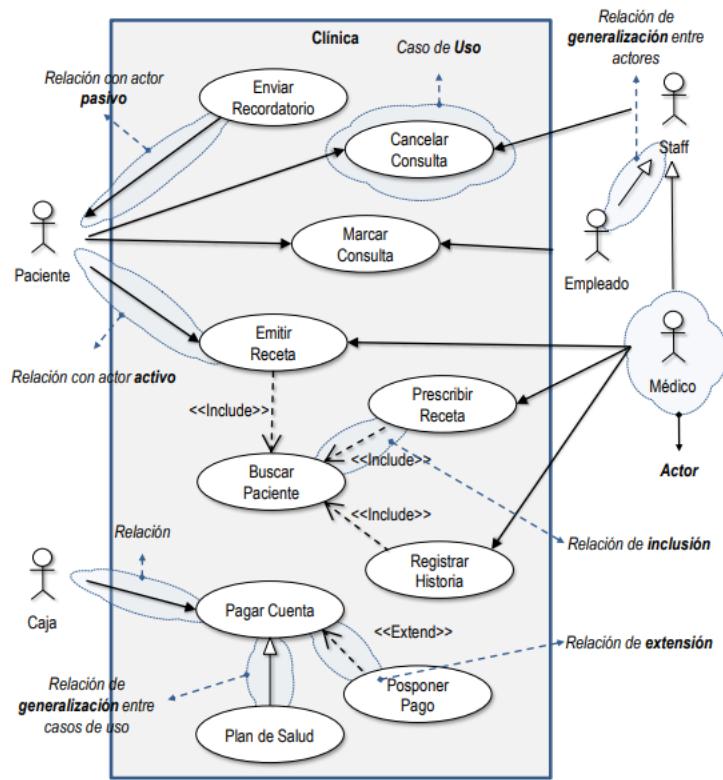


Figura 3.6: Ejemplo de diagrama de casos de uso para un sistema (complejo) en una clínica.

En este ejemplo se fuerza a hacer uso de todos los elementos de un diagrama de casos de uso. Normalmente no tienen por qué darse absolutamente todos los elementos en un mismo

diagrama de casos de uso.

Cabe añadir que se desecharon otros diagramas como los diagramas de secuencia [58] o los diagramas de estados [59] por claridad en el diseño y porque debido a la estructura de estos y del proyecto, tenía mucho más sentido usar los diagramas de casos de uso, pues son mucho más abstractos, sencillos e intuitivos.

### 3.2.1. Extracción de datos

Empezando por la extracción de datos, el diagrama de casos de uso resultante será el siguiente (3.7):

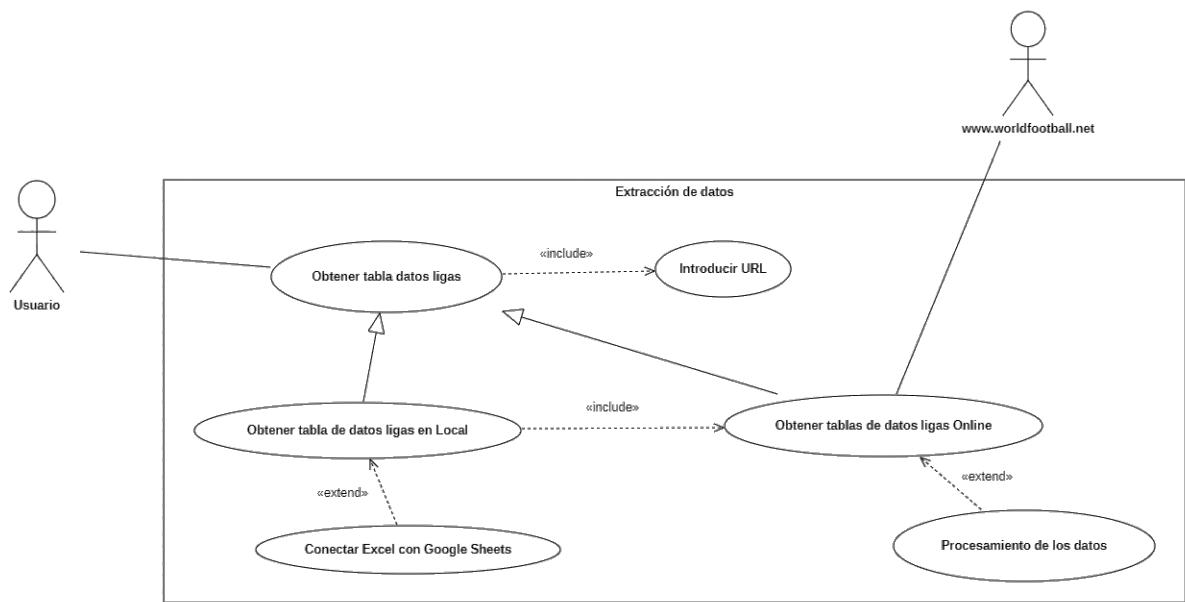


Figura 3.7: Diagrama de casos de uso para el sistema de extracción de datos.

En este diagrama encontramos 2 actores y 6 casos de uso.

El actor “Usuario” hace referencia a toda la gente que quiera obtenerse los datos de las ligas de fútbol usando este proceso, en este caso en concreto, serán el usuario como tal y el desarrollador, pero se han considerado ambos como usuarios pues aunque puedan tener distintos objetivos, ambos son usuarios de este sistema de extracción de datos. Por otro lado está el actor “www.worldfootball.net”, quien como se puede observar en la imagen, es el proveedor de los datos buscados.

Entrando en lo que es el diagrama, se observa que el actor usuario tiene solo una funcionalidad, la de obtenerse los datos de las ligas, aunque por la generalización, esta permitirá obtenérselos tanto de forma local como online. Si se continúa observando, se ve que para ello el usuario tendrá que introducir una URL antes, concretamente la de la liga en cuestión.

Una vez introducida la URL, si el usuario quisiera obtenerse los datos en local, antes tendría que obtenérselos de manera online antes por la relación de *<< include >>* que hay entre ambos casos de uso. Como se puede observar también, los *<< extend >>* añaden que para poder obtenerse estos datos de manera local, se puede conectar Excel a Google Sheets; y para obtenérselos de manera online se les puede dar un procesamiento para dejar en mejores condiciones a los datos provenientes desde [www.worldfootball.net](http://www.worldfootball.net) [18], en este caso el procesamiento será obligado, pero no es una función propia de obtención de los datos, por lo que se pone como *<< extend >>*.

### 3.2.2. Informatización de cálculos

A continuación se hará la informatización de los cálculos, dando como resultado el siguiente el diagrama de casos de uso (3.8):

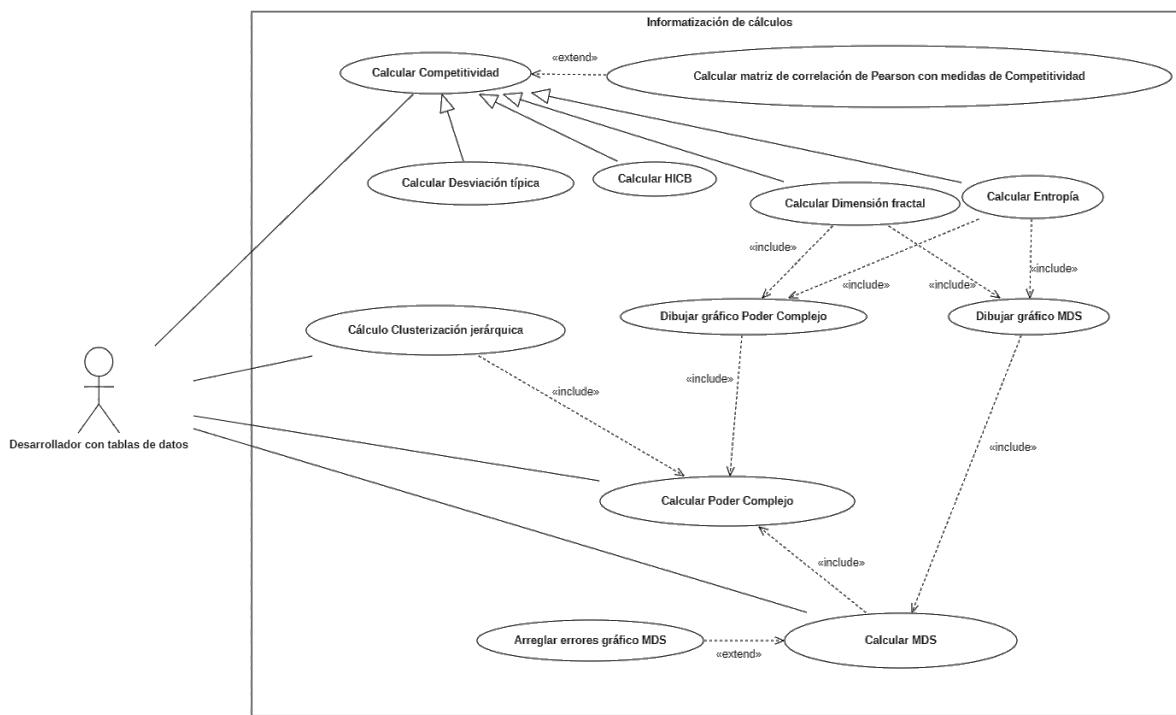


Figura 3.8: Diagrama de casos de uso para el sistema de informatización de cálculos.

donde se obtiene como resultado 1 solo actor y un total de 12 casos de uso diferentes.

En este caso el único actor es el desarrollador, quien tiene las tablas de datos de las ligas obtenidas del proceso de extracción de datos anterior. Esta vez si que se pone desarrollador y no usuario, pues los cálculos que hay por detrás solo le serán útiles a este desarrollador, el usuario final se pretende que solo los visualice.

En primer lugar tendrá que obtenerse el poder complejo, pues como podemos observar por los <<include>>, casi todos los casos de uso dependen de este primer cálculo. A parte del cálculo del poder complejo, el desarrollador deberá poder hacer una clusterización jerárquica, el cálculo de la competitividad y el cálculo del MDS, el cual se podría dejar tal cual se obtiene, pero se le añade la corrección de errores como extensión pues no es una funcionalidad propia del cálculo del MDS.

Observando el cálculo de la competitividad, se ve que la competitividad hace referencia y está formada por las 4 medidas estudiadas en el artículo [1]. La desviación típica y el *HICB* no necesitarán nada más que los datos de las ligas para ser calculados, sin embargo, los cálculos de la entropía y de la dimensión fractal se obtienen a partir de los gráficos del poder complejo y del MDS, por lo que ambos gráficos deberán ser calculados antes, es por esto que se queda esta cadena de <<include>> en el diagrama final. El cálculo de la matriz de correlación está puesto como extensión pues es algo que se puede sacar o no de las medidas, pero no calcula la competitividad de estas como tal, sino la relación entre ellas.

### 3.2.3. Visualización de resultados

En último lugar, para visualizar los resultados obtenidos, se seguirá el siguiente diagrama de casos de uso (3.9):

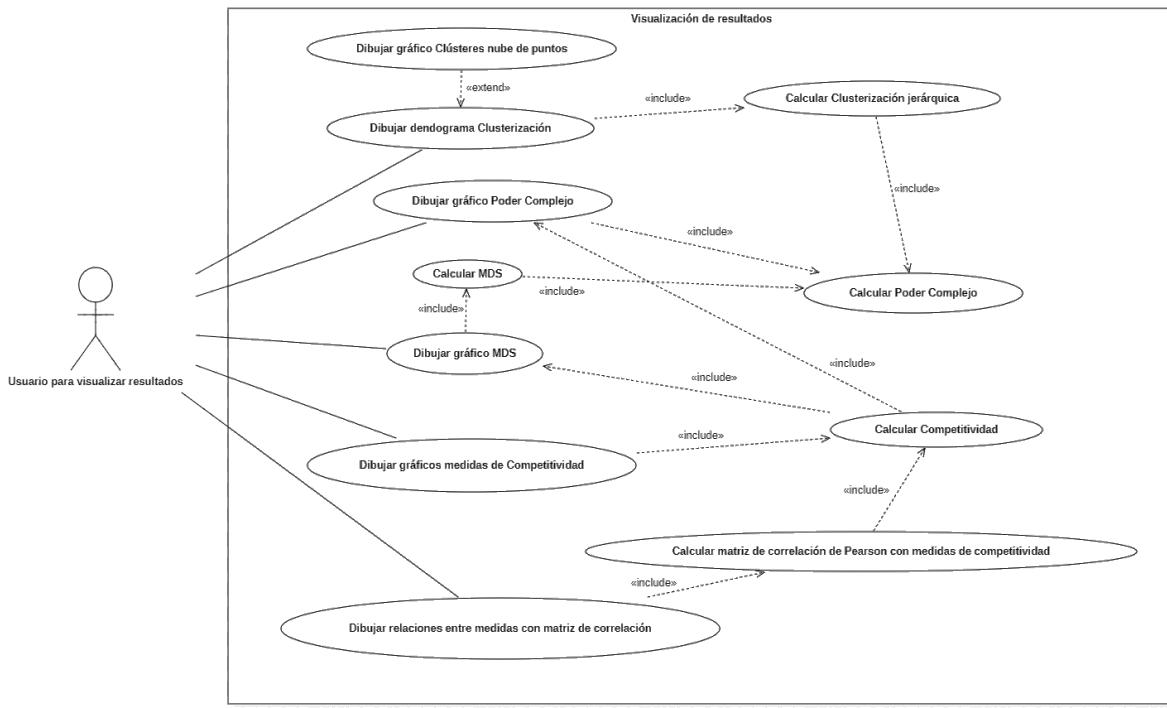


Figura 3.9: Diagrama de casos de uso para el sistema de visualización de resultados.

Aquí tenemos de nuevo solo 1 actor y un total de 11 casos de uso.

El actor en este caso es el usuario que quiere visualizar los resultados, por lo que su objetivo final es únicamente el de obtenerse unos gráficos explicativos de cada uno de los cálculos anteriores. Es por esto que tiene un total de 5 funcionalidades a las que puede acceder directamente que le dibujan los gráficos que deseé.

Se observa de nuevo que lo primordial es calcular el poder complejo, pues lo requieren la mayoría de cálculos. A pesar de que esta fase no es de cálculos, evidentemente para dibujar los resultados antes hay de obtenérselos, por ello se repiten muchas estructuras y casos de uso vistos en el punto 3.2.2 anterior.

Una diferencia entre el diagrama de la informatización de cálculos 3.8 anterior y este, es que entre el cálculo de la competitividad y el cálculo de la matriz de correlación, su relación pasa de *<< extend >>* a ser de *<< include >>*, pues ahora el dibujo de las relaciones es una funcionalidad principal del sistema y requiere del cálculo de la competitividad para ello, mientras que antes (en la sección 3.2.2) era una funcionalidad extra que aparecía una vez calculada la competitividad.

En cuanto a la única relación de *<< extend >>* que hay, esta permite dibujar gráficos con

una nube de puntos para los clústeres. Es de <<extend>> pues es una funcionalidad extra una vez obtenidos los clústeres con el dendrograma. Además, se quiere que no se puedan obtener este tipo de gráficos sin que se hayan dibujado sus respectivos dendrogramas antes.

Por último añadir que no se dibuja el mismo desglose para el caso de uso de calcular la competitividad visto en la diagrama de la informatización de cálculos (imagen 3.8) únicamente por claridad en el diagrama resultante obtenido aquí (imagen 3.9), pero la misma generalización se mantiene aunque no esté dibujada.

### 3.3. Implementación

En el ciclo de implementación se profundizará sobre las funciones creadas y no solo sus usos, sino en detalle qué hacen y cómo. Se evitará entrar en demasiada profundidad para evitar la sobreexplicación, pero sin dejar de lado el cómo y el por qué de la implementación [44]. Este ciclo explicará algunos de los cálculos del artículo original [1], con el fin de clarificar la implementación final.

#### 3.3.1. Extracción de datos

En cuanto a la implementación de la extracción de datos se describirá cómo se obtienen a partir de los diferentes links o URLs. Estos datos serán los puntos de todas las jornadas que tenga cada liga. Es decir, en el caso de la liga italiana durante la temporada 2018/2019, se necesitarán los datos de 38 links diferentes pues contaba con 38 jornadas en total. Una vez obtenidos estos datos se explicará cómo se procesan para ser legibles. Cuando ya estén procesados se le pondrá un proceso de automatización para no necesitar supervisión. En último lugar, se explicará cómo se conecta una hoja de Excel a Google Sheets para que se obtenga los datos automáticamente.

En primer lugar se usará la función *ImportXML* [20] como se menciona en la sección 2.2.1. Esta necesita de un link y de una etiqueta XML de la que obtener los datos. La etiqueta en cuestión en este caso será: `// *[@id = "site"] / div[2] / div[1] / div[1] / div[7] / div / table[1] / tbody`, la cual devolverá todos los equipos de esa jornada con un equipo en cada celda y todos sus datos. Estos requerirán un procesamiento que se verá más tarde. La etiqueta se obtiene como se explica en la sección 2.2.1: pinchando en inspeccionar a un elemento de una página web (o “Ctrl + Mayús + I”), botón derecho sobre la etiqueta de la que se quieran sus datos, por

ejemplo en la tabla de los datos de la liga, “Copy > Copy XPath”, y, en este caso, se obtiene “// \*[@id = ”site”]/div[2]/div[1]/div[1]/div[7]/div/table[1]/tbody” como resultado.

Puesto que se necesita un link por jornada, se llegarán a tener que obtener los datos de 38 links diferentes. Esto no es demasiado problema, pues la única diferencia entre el link de la jornada 1: <https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/1/>, y el link de la jornada 38: <https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/38/>, es el cómo acaba cada uno, por lo que será sencillo poner un link por fila. Se pondrán un total de 40 links para cubrir el caso de que hubiera 40 jornadas en total, aunque lo máximo que se han visto son 38. Por suerte poner un link como <https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/40/> en el *ImportXML* no devolverá nada, pues esa jornada no existe.

La diferencia entre los links de una liga y otra está en el cómo viene dada cada una, por ejemplo, para la liga española es tal que: <https://www.worldfootball.net/schedule/esp-primeradivision-2018-2019-spieltag/1/>. Por esto los Id's de ligas correspondientes que se le exigen introducir al usuario son los siguientes en este caso (imagen 3.10 tomada del proyecto desarrollado final en Google Sheets llamado “Rastrealigas” [45]):

| # (1-7) | ID_Liga                       | URL   |
|---------|-------------------------------|---|
| 1       | esp-primeradivision-2018-2019 | <a href="https://www.worldfootball.net/schedule/esp-primeradivision-2018-2019-spieltag/">https://www.worldfootball.net/schedule/esp-primeradivision-2018-2019-spieltag/</a> |
| 2       | eng-premier-league-2018-2019  | <a href="https://www.worldfootball.net/schedule/eng-premier-league-2018-2019-spieltag/">https://www.worldfootball.net/schedule/eng-premier-league-2018-2019-spieltag/</a>   |
| 3       | ita-serie-a-2018-2019         | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/">https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/</a>                 |
| 4       | fra-ligue-1-2018-2019         | <a href="https://www.worldfootball.net/schedule/fra-ligue-1-2018-2019-spieltag/">https://www.worldfootball.net/schedule/fra-ligue-1-2018-2019-spieltag/</a>                 |
| 5       | bra-serie-a-2019              | <a href="https://www.worldfootball.net/schedule/bra-serie-a-2019-spieltag/">https://www.worldfootball.net/schedule/bra-serie-a-2019-spieltag/</a>                           |
| 6       | jpn-j1-league-2019            | <a href="https://www.worldfootball.net/schedule/jpn-j1-league-2019-spieltag/">https://www.worldfootball.net/schedule/jpn-j1-league-2019-spieltag/</a>                       |
| 7       | bundesliga-2018-2019          | <a href="https://www.worldfootball.net/schedule/bundesliga-2018-2019-spieltag/">https://www.worldfootball.net/schedule/bundesliga-2018-2019-spieltag/</a>                   |

Figura 3.10: Id de todas las ligas a analizar con sus respectivos URLs.

Una vez hecho esto, se ponen sus 40 links correspondientes en el capturador de los datos, quedando tal que así (imagen 3.11 tomada del Google Sheets “Rastrealigas” [45]):

| # | URL_IDLiga  | Jornada | URL   | DatPrinc01   | DatPrinc02 | DatPrinc03 | DatPrinc04 | DatPrinc05 | DatPrinc06 |
|---|---|---------|---|--|------------|------------|------------|------------|------------|
| 3 | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | 1       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Atalanta 1100 2 Empoli FC 110 3 Juventus 1100 4 AC Milan 1100 SSC Napoli 1000   |            |            |            |            |            |
|   |   | 2       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 2200 2 SSC Napoli 22 3 SPAL 2013 Fe 4 Atalanta 2110 5 AS Roma 2100   |            |            |            |            |            |
|   |   | 3       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 3300 2 Sassuolo Calc 3 ACF Fiorentina 4 AC Milan 3201 5 SPAL 2100  |            |            |            |            |            |
|   |   | 4       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 4400 2 SPAL 2013 Fe 3 SSC Napoli 434 5 Sampdoria 42 5 Sassuolo Calc 3 ACF Fiorentina 4 AC Milan 3201 5 SPAL 2100 |            |            |            |            |            |
|   |   | 5       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 5500 2 SSC Napoli 543 5 Sassuolo Calc 4 ACF Fiorentina 5 Lazio 6 Roma 6 5 Inter 63                               |            |            |            |            |            |
|   |   | 6       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 6600 2 SSC Napoli 65 3 Sassuolo Calc 4 Lazio 5 Roma 6 5 Inter 63   |            |            |            |            |            |
|   |   | 7       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 7700 2 SSC Napoli 75 3 Inter 74120 5 4 Sassuolo Calc 5 ACF Fiorentina 6 Roma 6 5 Inter 63                        |            |            |            |            |            |
|   |   | 8       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 8800 2 SSC Napoli 86 3 Inter 85120 6 4 AC Milan 8431 5 Lazio Roma 6 5 Inter 63                                   |            |            |            |            |            |
|   |   | 9       | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 9810 2 SSC Napoli 97 3 Inter 96123 6 4 Lazio Roma 9 5 Sampdoria 10 5 Inter 63                                    |            |            |            |            |            |
|   |   | 10      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1091 2 Inter 1071216 3 SSC Napoli 104 4 AC Milan 1055 5 Lazio Roma 10 5 Inter 63                                 |            |            |            |            |            |
|   |   | 11      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1110 2 Inter 1181221 3 SSC Napoli 114 4 AC Milan 1163 5 Lazio Roma 11 5 Inter 63                                 |            |            |            |            |            |
|   |   | 12      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1211 2 SSC Napoli 123 3 Inter 1281322 4 Lazio Roma 11 5 AC Milan 1200 5 Inter 63                                 |            |            |            |            |            |
|   |   | 13      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1312 2 SSC Napoli 133 3 Inter 1391325 4 Lazio Roma 11 5 AC Milan 1300 5 Inter 63                                 |            |            |            |            |            |
|   |   | 14      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1413 2 SSC Napoli 143 3 Inter 1492327 4 AC Milan 1474 5 Lazio Roma 1400 5 Inter 63                               |            |            |            |            |            |
|   |   | 15      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1514 2 SSC Napoli 15 3 Inter 1592427 4 AC Milan 1575 5 Lazio Roma 1500 5 Inter 63                                |            |            |            |            |            |
|   |   | 16      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1615 2 SSC Napoli 16 3 Inter 1610242 4 AC Milan 1676 5 Lazio Roma 1600 5 Inter 63                                |            |            |            |            |            |
|   |   | 17      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1716 2 SSC Napoli 17 3 Inter 1710342 4 Lazio Roma 11 5 AC Milan 1700 5 Inter 63                                  |            |            |            |            |            |
|   |   | 18      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1816 2 SSC Napoli 18 3 Inter 1811343 4 Lazio Roma 11 5 Sampdoria 1800 5 Inter 63                                 |            |            |            |            |            |
|   |   | 19      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 1917 2 SSC Napoli 19 3 Inter 1912343 4 Lazio Roma 11 5 AC Milan 1900 5 Inter 63                                  |            |            |            |            |            |
|   |   | 20      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 2018 2 SSC Napoli 20 3 Inter 2012443 4 AC Milan 2097 5 AS Roma 2000 5 Inter 63                                   |            |            |            |            |            |
|   |   | 21      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 2119 2 SSC Napoli 21 3 Inter 211245314 4 AC Milan 2198 5 AS Roma 2100 5 Inter 63                                 |            |            |            |            |            |
|   |   | 22      | <a href="https://www.worldfootball.net/schedule/ita-serie-a-2">https://www.worldfootball.net/schedule/ita-serie-a-2</a> | #TeamM.WDLgc 1 Juventus 2219 2 SSC Napoli 22 3 Inter 2212463 4 AC Milan 2295 5 Atalanta 2200 5 Inter 63                                  |            |            |            |            |            |

Los equipos y sus puntuaciones vienen ordenados en función de la posición de la tabla en esa jornada y no en función del nombre de los equipos, es decir, el orden de los equipos es distinto en la jornada 1 que en la jornada 17. Por ello habrá que almacenar qué equipo tiene qué puntos en cada momento de la temporada, para ello se les asocia una posición correspondiente. La idea es la que se ve a continuación (imagen 3.12 tomada del Google Sheets “Rastrealigas” [45]):

| Equipo_01 | Equipo_02       | Equipo_03       | Equipo_04       | Equipo_05       | Equipo_06       | Equipo_07       | Puntos_01 | Puntos_02 | Puntos_03 | Puntos_04 | Puntos_05 | Puntos_06 | Puntos_07 |
|-----------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| Atlanta   | Empoli FC       | Juventus        | AC Milan        | SSC Napoli      | AS Roma         | Sassuolo Calcio | 3         | 3         | 3         | 3         | 3         | 3         | 3         |
| Juventus  | SSC Napoli      | SPAL Ferrara    | Atalanta        | AS Roma         | Sassuolo Calcio | Udinese Calcio  | 6         | 6         | 6         | 4         | 4         | 4         | 4         |
| Juventus  | Sassuolo Calcio | ACF Fiorentina  | AC Milan        | SPAL Ferrara    | SSC Napoli      | Atalanta        | 9         | 7         | 7         | 6         | 6         | 6         | 4         |
| Juventus  | SPAL Ferrara    | SSC Napoli      | Sampdoria       | Sassuolo Calcio | AC Milan        | ACF Fiorentina  | 12        | 9         | 9         | 7         | 7         | 7         | 7         |
| Juventus  | SSC Napoli      | Sassuolo Calcio | ACF Fiorentina  | Lazio Roma      | SPAL Ferrara    | Udinese Calcio  | 15        | 12        | 10        | 10        | 9         | 9         | 8         |
| Juventus  | SSC Napoli      | Sassuolo Calcio | Lazio Roma      | Inter           | ACF Fiorentina  | AC Milan        | 18        | 15        | 13        | 12        | 10        | 10        | 9         |
| Juventus  | SSC Napoli      | Inter           | Sassuolo Calcio | ACF Fiorentina  | AC Milan        | Lazio Roma      | 21        | 15        | 13        | 13        | 13        | 12        | 12        |
| Juventus  | SSC Napoli      | Inter           | AC Milan        | Lazio Roma      | Sampdoria       | AS Roma         | 24        | 18        | 16        | 15        | 15        | 14        | 14        |
| Juventus  | SSC Napoli      | Inter           | Lazio Roma      | Sampdoria       | AC Milan        | AS Roma         | 25        | 21        | 19        | 18        | 15        | 15        | 14        |
| Juventus  | Inter           | SSC Napoli      | AC Milan        | Lazio Roma      | Sampdoria       | AS Roma         | 28        | 22        | 22        | 18        | 18        | 15        | 15        |
| Juventus  | Inter           | SSC Napoli      | AC Milan        | Lazio Roma      | Sassuolo Calcio | Torino FC       | 31        | 25        | 25        | 21        | 21        | 18        | 17        |
| Juventus  | SSC Napoli      | Inter           | Lazio Roma      | AC Milan        | AS Roma         | Sassuolo Calcio | 34        | 28        | 25        | 22        | 21        | 19        | 19        |
| Juventus  | SSC Napoli      | Inter           | Lazio Roma      | AC Milan        | Parma Calcio    | AS Roma         | 37        | 29        | 28        | 23        | 22        | 20        | 19        |
| Juventus  | SSC Napoli      | Inter           | AC Milan        | Lazio Roma      | Torino FC       | AS Roma         | 40        | 32        | 29        | 25        | 24        | 21        | 20        |
| Juventus  | SSC Napoli      | Inter           | AC Milan        | Lazio Roma      | Torino FC       | Atalanta        | 43        | 35        | 29        | 26        | 25        | 22        | 21        |
| Juventus  | SSC Napoli      | Inter           | AC Milan        | Lazio Roma      | Atalanta        | AS Roma         | 46        | 38        | 32        | 27        | 25        | 24        | 24        |
| Juventus  | SSC Napoli      | Inter           | Lazio Roma      | AC Milan        | Sampdoria       | Sassuolo Calcio | 49        | 41        | 33        | 28        | 27        | 26        | 25        |

Figura 3.12: Equipos indexados.

Esto permitirá saber qué puntos corresponden a qué equipo únicamente con desplazarse un número fijo de celdas a la derecha.

El problema principal es que los números vienen todos juntos y solo se quiere obtener el

dato final, que son los puntos de ese equipo en ese momento de la temporada. Observando la siguiente imagen (imagen 3.13 tomada de <https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/1/>) se ve que hay que encontrar una manera para obtenerse los puntos del equipo a pesar de estar todos los datos juntos sin ninguna manera clara de diferenciar a qué corresponde cada número a priori.

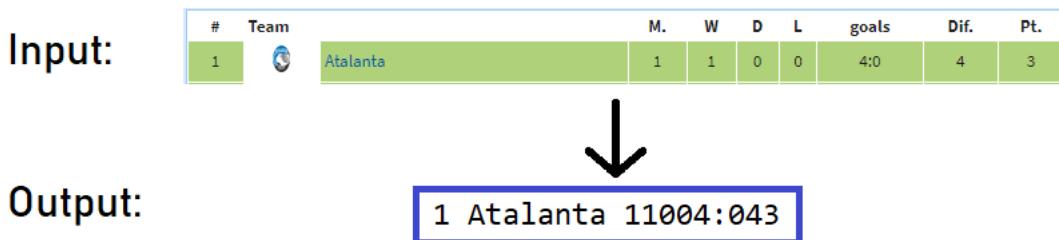


Figura 3.13: Ejemplo de output del *ImportXML*.

Para obtenerse los nombres de los equipos se quitarán todos los “-”, “:” y números del *string* de los datos en bruto, por lo que en ocasiones se perderá algún detalle en equipos como el “Schalke 04”, que quedará como “Schalke” solamente.

Para resolver el problema de cómo obtenerse los puntos, se observa que de lo único que se puede estar seguro es de que el carácter que está más a la derecha del *string* de los datos en bruto corresponde a la cifra de las unidades de los puntos en esa jornada. Así que para la primera jornada, lo que se hace es obtenerse esta cifra que está a la derecha del todo.

Para las siguientes jornadas, se conocerán de nuevo las unidades de los puntos buscados, pero ahora se tendrán también los puntos de la jornada anterior. La idea es por tanto ver si cambia la cifra de las decenas, y de cambiar las decenas, hay que ver si cambia el número de caracteres que hay que obtenerse, pues no es lo mismo que cambien los puntos de 9 a 12 (se pasa de coger 1 carácter a coger 2), que si cambian de 29 a 32 (se cogen 2 caracteres en ambos casos).

Para ello hay que fijarse en que las cifras de las unidades. En el caso en el que las unidades de la jornada actual sean mayores o iguales que los de la jornada anterior, se cogerán el mismo número de caracteres que en la jornada anterior. Esto es así porque los puntos avanzan como mucho de 3 en 3, nunca 10 o más. Es decir, si se tienen 13 puntos en la jornada anterior, y se ha visto que las unidades de la nueva jornada tienen 6, esto significa que como  $6 \geq 3$ , entonces la cifra de las decenas no ha cambiado, por lo que se cogen el mismo número de caracteres que en

la jornada anterior, en este caso 2 caracteres porque 13 tiene 2 caracteres en total, obteniéndose así 16 puntos como resultado.

El problema principal ocurre si las unidades de la jornada actual son menores que las de la jornada anterior, pues esto significa que ha habido un cambio en las decenas. Es por esto que se observa la cantidad total de puntos de la jornada anterior, pues si es mayor o igual a 97, habrá que cogerse 3 caracteres en vez de 2 (se ha pasado de 98 a 101 por ejemplo) y si está entre 9 y 7 se tendrán que coger 2 caracteres en vez de 1 (se ha pasado de 9 a 10 por ejemplo). Se cogen 7 y 97 porque como se ha mencionado antes, los puntos en las ligas de fútbol van de 3 en 3 como mucho.

La forma de saber si se tienen que comprobar con 7 o 97 (o incluso 997), es con la siguiente fórmula (3.1):

$$\text{Comprobador de decenas} = 10^{\text{caracteres puntuación jornada anterior}} - 3. \quad (3.1)$$

Con ella si por ejemplo  $\text{caracteres puntuación jornada anterior} = 2$ , entonces la puntuación de la jornada anterior se verá si es  $\geq 97$ ; y si  $\text{caracteres puntuación jornada anterior} = 1$ , si es  $\geq 7$ .

Este problema se resolverá de forma más elegante con las funciones creadas en Javascript para Google Sheets llamadas “checkPuntosIni”, que se recibe un *string* con los datos del equipo tal cual vienen del *ImportXML* (como se ven en la imagen 3.13) y se coge el carácter de la derecha del todo que corresponde a la cifra de las unidades de la jornada 1; y “checkeaPuntos”, que se recibe este mismo *string* y además un *integer* de los puntos en esa posición de la jornada anterior. La implementación final de ambas funciones no se pondrá en la memoria del TFG por comodidad en la lectura, pero de todos modos sigue lo aquí descrito.

En principio esta implementación para solucionar este problema funciona perfectamente, pero como se verá en la sección 3.4.1, surge un inconveniente si puede haber puntuaciones negativas. Esto puede ocurrir y ha ocurrido si a algún equipo se le sanciona antes de empezar la competición, igual que le ocurrió al Chievo Verona en la liga italiana durante la temporada 2018/2019 [60], una de las 7 temporadas analizadas en este TFG. La implementación que se sigue en el caso de las puntuaciones negativas es similar, pero de todos modos se explicará en la sección de pruebas 3.4.1.

Una vez obtenidos los datos de una liga, se hace uso de las funciones *Índice* y *Coincidir*, que son muy similares a *BuscarV*. Estas buscarán los equipos en la jornada correspondiente y pondrán los puntos que tenía dicho equipo según la asociación vista en la imagen 3.12.

Una vez obtenidos los datos como se necesitan, para que se actualice (guardar los datos y pasar a la siguiente liga) de forma automática habrá que hacer una función que compruebe que ya se ha obtenido todo de esa liga, y de ser así, copiar los datos que se quieren obtener, pegarlos en un almacén de datos (la hoja “Almacen” en este caso) y pasar a la URL de la siguiente liga para obtenerse los datos de esta.

La función encargada de esto será la llamada “cargaLigas”. Esta se comprobará si se han obtenido ya todos los datos en función del número de casillas en las que aún ponga “Loading...”. Si ve que ya se han cargado, copia y pega los datos en la hoja “Almacen” y aumenta en 1 los índices. Si está en el último índice para y deja de aumentar.

Para que esta función se ejecute de manera automática y sin revisión se le colocará un disparador que la ejecute cada minuto. Una vez obtenidos los datos requeridos podremos quitar este disparador, pero puesto que si se ejecuta una vez llegado al final no ocurre nada, no es de demasiada importancia. Aún así se recomienda quitarlos una vez terminada la extracción de datos, pues como se menciona anteriormente, Google Sheets se ejecuta en segundo plano [21], así que aunque esté el navegador cerrado, el disparador seguirá ejecutando la función “cargaLigas”.

Por último, una vez obtenidos todos los datos como se desean, se conectarán con un fichero Excel en local. Para ello habrá que ir a Google Sheets y pinchar en: *Archivo > Publicar en la Web → Enlace > seleccionar la Hoja (Almacen) > .csv > Publicar* → Copiar y guardar ese enlace.

Una vez hecho esto, hay que abrir un Excel nuevo y clickar en: *Datos > Obtener datos > Desde otra fuente > Desde la web → Básico > Pegar el link copiado en el paso anterior > Aceptar* → *Origen del archivo : “65001 : Unicode (UTF – 8)” > Delimitado : “Coma” > Detección del tipo de datos : “Basado en todo el conjunto de datos” > Cargar*.

Esto nos permitirá tener los datos en local como en la imagen 3.2. Si queremos que estos datos se actualicen cada vez que se abre el archivo por ejemplo, habrá que darle a: *Datos > Actualizar todo > Propiedades de conexión... → Uso > Actualizar al abrir el archivo > Aceptar*.

Esto dará como resultado final un archivo en Excel al que se le ha llamado “Conexion\_DatosLigas”, el cual se actualiza cada vez que se abre en función de lo que tenga el almacén (la hoja “Alma-

cen”) del Google Sheets “RastreaLigas” en ese momento. Si se quieren actualizar los datos en local de forma manual siempre se puede pulsar el botón que está encima de *Actualizar todo*.

### 3.3.2. Informatización de cálculos

Una vez obtenidos todos los datos, se procederá con la informatización de los cálculos haciendo uso de ellos. Para ello se hará uso de múltiples librerías, en concreto las mencionadas en la sección 2.2.2.

En primer lugar, para leer los datos del Excel se hace uso de la función llamada “*getDataExcel*”. Esta comienza usando la función *read\_excel* [61] de la librería *pandas*, con la que se leen los datos del Excel del que se le pase la ruta y lo deja como un gran *Dataframe* (explicados en la sección 2.2.2) ignorándose las primeras 10 columnas del Excel, pues contienen datos extras que no son las tablas descritas en la sección 3.3.1 anterior. Esto deja las 7 tablas a analizar como 1 solo *Dataframe*, por ello se utilizará una variable auxiliar que pondrá 1’s en todas las celdas con datos y 0’s en las que ponga “*Nan*”. De esta forma se obtendrá una lista de *Dataframes*, en concreto 7 en este caso, pues había 7 tablas en total. Los *Dataframes* resultantes serán transformados a *numpy arrays* de 2 dimensiones (un *numpy arrays* funciona igual que un *array*, pero de esta librería, por lo que tiene más funciones). La solución está basada en la referencia [62].

Como se ha visto en la sección de Diseño 3.2.2, lo primordial es calcular el poder complejo. Para ello no hay más que recorrerse el *array* por filas, es decir, por equipos, fijando siempre cuál era el valor anterior y en función de los puntos ganados, ir asignándole unos puntos u otros en función de si el equipo ganó, empató o perdió. Esto significa que si ganó (*puntos ganados == 3*) se le suma a la posición anterior (3,0), si empató (*puntos ganados == 1*) se le suma (1,2) a esta posición y si perdió (*puntos ganados == 0*) se le suma (0,3) a la posición anterior también. Para el caso base, es decir, para la jornada 0, todos los equipos empiezan desde (0,0).

Para que este cálculo del poder complejo funcione se necesita una copia de los datos de los puntos a la que ir asignando todas estas posiciones. De esta manera se podrán recorrer ambas matrices a la vez, la matriz de datos, la cual se lee, y la matriz de resultados, en la que se escribe en función de lo leído. Para hacer esta copia de las matrices será necesario usar la función *copy.deepcopy* de la librería *copy* [63]. El problema de hacer la copia con una simple asignación es que *Python* no hace una copia real de la variable, solamente le da otro nombre a esa misma variable. No solo esto, sino que existía la opción de usar la función *copy.copy* en

vez de *copy.deepcopy*, la cual sí que hacía una copia de la variable, pero solo de profundidad 1 [64]. Es por todo esto que se opta por *copy.deepcopy*.

En la siguiente imagen se ve una aclaración de estas funciones (imágenes 3.14 editadas, pero tomadas de [64]):

#### Asignación:

```
old_list = [[1, 2, 3], [4, 5, 6], [7, 8, 'a']]
new_list = old_list

new_list[2][2] = 9

print('Old List:', old_list)
print('ID of Old List:', id(old_list))

print('New List:', new_list)
print('ID of New List:', id(new_list))
```

#### Output:

```
Old List: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
ID of Old List: 140673303268168

New List: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
ID of New List: 140673303268168
```

#### "*copy.copy*":

```
import copy

old_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]]
new_list = copy.copy(old_list)

old_list[1][1] = 'AA'

print("Old list:", old_list)
print("New list:", new_list)
```

#### Output:

```
Old list: [[1, 1, 1], [2, 'AA', 2], [3, 3, 3]]
New list: [[1, 1, 1], [2, 'AA', 2], [3, 3, 3]]
```

#### "*copy.deepcopy*":

```
import copy

old_list = [[1, 1, 1], [2, 2, 2], [3, 3, 3]]
new_list = copy.deepcopy(old_list)

old_list[1][0] = 'BB'

print("Old list:", old_list)
print("New list:", new_list)
```

#### Output:

```
Old list: [[1, 1, 1], ['BB', 2, 2], [3, 3, 3]]
New list: [[1, 1, 1], [2, 2, 2], [3, 3, 3]]
```

Figura 3.14: Diferencias entre asignación, *copy.copy* y *copy.deepcopy*.

El método *copy.deepcopy* se utilizará cada vez que se quiera trabajar con la lista de ligas, así no se sobrescribirán datos de utilidad y relevantes para el futuro.

Una vez calculado el poder complejo ( $S_i$ ), se procederá a calcular el dendrograma y la nube de puntos resultantes del proceso de clusterización o agrupación. Para esto solo se necesita la posición final del poder complejo, es decir  $S_i(R)$ , donde  $R$  hace referencia a la última ronda o jornada.

Para el cálculo del dendrograma se hará uso de las funciones *linkage* [32] y *dendrogram* [31] de la librería *scipy.cluster.hierarchy* [65].

La función *linkage* necesita únicamente una lista de puntos, el método de agrupamientos con el que se medirá y la métrica de distancias a seguir. Como uno de los objetivos finales es el de comparar el resultado final con los resultados obtenidos por parte del artículo: “Modeling and visualizing competitiveness in soccer leagues” [1], se seguirán los mismos métodos y métricas que este use. En este caso, el método usado por el artículo [1] es el del agrupamiento de enlace media o promedio [3], el cual se asigna con el parámetro “method” tal que: “method=‘average’”; y la métrica para las distancias que se emplea en el mismo artículo [1] es la distancia euclídea-na (aunque podría ser la conocida como “cityblock” según la interpretación dada como se verá en el capítulo 5) [4], asignada con: “metric=‘euclidean’ ”. En cualquier caso, estas métricas y métodos se pondrán en duda en el capítulo 5 por los resultados obtenidos.

Y para el cálculo del dendrograma se hará uso de la función *AgglomerativeClustering* [37] de la librería *sklearn.cluster* [66]. Esta funciona exactamente igual que la anterior, pero se utiliza “linkage=‘average’ ” para el “method”; y “affinity=‘euclidean’ ” para la “metric”. Además de esto, se marcará el número de clústeres o grupos que se desean con el parámetro “n\_clusters”. Como el artículo [1] no aclara por qué en algunas ocasiones coge 4 grupos y en otras 5, se le pondrán “n\_clusters=5”, es decir, 5 grupos a todas las ligas para que sea más robusto y consistente. Añadir que como bien dice el nombre de la función, la clusterización será aglomerativa, que es la que se busca en este caso.

Tras esto, se procederá al cálculo del MDS. Para ello lo que hay que hacer es transformar la matriz de puntos del poder complejo anterior a una lista de puntos, con esta lista de puntos obtenerse una matriz de distancias usando la distancia de Manhattan o “cityblock” (explicada en el artículo [1] y en el TFG, pero de forma simplificada, es una distancia que utiliza la suma de los catetos en vez de la hipotenusa [4]) y una vez se tiene esta matriz, aplicar el algoritmo

de escalamiento multidimensional o MDS, el cual transformará una matriz de distancias a una matriz de puntos (puntos de 2 dimensiones en este caso). La solución está basada en la referencia [67].

Para transformar la matriz (*array* de 2 dimensiones) de puntos a una lista de puntos, basta con utilizar *.flatten().tolist()*, donde *.flatten()* [68] pasa el *array* de 2D a 1D, y *.list()* [69] lo pasa de un *array* a una lista. Hecho esto, solo hace falta usar la función *pairwise\_distances* con esta lista en la *X* y la *Y*, poniendo ahora si, la distancia de Manhattan como métrica con “metric='cityblock'”.

Una vez se tiene una matriz de distancias que llega a ser de dimensiones  $780 \times 780$  (*jornadas·equipos* =  $(38 + 1) \cdot 2 = 780$ ), se le aplica el algoritmo de MDS para que vuelva a dejarlo como una matriz de puntos equivalente a la del poder complejo, pero distinta. Esto se hará haciendo uso de la función *sklearn.manifold.MDS* [42] de la librería *sklearn* [34]. Los parámetros que serán relevantes son: “n\_components = 2”, “metric=True”, “random\_state=5” y “dissimilarity = ‘precomputed’”. Los “n\_components = 2” porque se quiere que el escalamiento multidimensional sea a 2 dimensiones como hace el artículo [1]; “metric=True” pues es necesario que sea un escalamiento métrico, sino el resultado muy diferente al buscado; “random\_state=5” para evitar que salga un resultado diferente a cada ejecución (podría ser 5 como cualquier otro número); y “dissimilarity = ‘precomputed’”, pues como se ha dicho anteriormente, la distancia entre los puntos ya ha sido calculada.

El algoritmo de MDS dará problemas en su solución principalmente debido al factor aleatorio que tiene por usar un algoritmo de minimización, dejando algunos puntos prácticamente en el lado opuesto al que deberían. Este error no se observó hasta la visualización, pero se explicará y desarrollará sus soluciones en la sección de pruebas de la informatización de cálculos 3.4.3 y los resultados con errores se mostrarán en la sección de pruebas de la visualización de resultados 3.4.2.

Tras esto, se debe calcular la dimensión fractal, pero para ello se han de visualizar los gráficos antes, así que esa parte en concreto corresponderá a la sección 3.3.3, pero todo lo referido a los cálculos de la dimensión fractal de una imagen se verán a continuación. En este caso se sigue una implementación parecida a la descrita en el artículo [1], donde se realiza un conteo de cajas tocadas por los píxeles negros de la imagen y estas cajas se reducen a la mitad con cada iteración hasta llegar al mínimo que admite la imagen. La idea aquí seguida es básicamente la

misma, lo único que varía es que el número de iteraciones se calcula antes en este caso, en vez de hacerse hasta llegar a un valor. La implementación de la solución está basada en la referencia [70].

En cuanto al cálculo de la entropía le ocurre lo mismo que a la dimensión fractal respecto a sus gráficos. Para este cálculo de la entropía se hace uso de los “glcm”. Los “glcm” hacen uso de la función *graycomatrix* [71], la cual indica con un número del 1 al 8 (no del 1 al 256 por estar en escala de grises), la intensidad de un gris en cada casilla. El “glcm” cuenta el número de veces que se dan las intensidades a pares como se ve en la imagen 3.15. Una vez obtenido esto, será lo que se use como  $P_{xy}$  en la fórmula (3.2) de la entropía (H):

$$H = \sum_{x=1}^{N_x} \sum_{y=1}^{N_y} -P_{xy} \log P_{xy}; \rightarrow -np.sum(glcm * np.log2(glcm + (glcm == 0))), \quad (3.2)$$

pues  $P_{xy}$  se refiere a esta intensidad de píxeles negros a pares que aporta el “glcm”. La implementación de esta solución está basada en la referencia [72].

Un ejemplo de cómo se obtiene un “glcm” a partir de una matriz de grises será el siguiente (imagen tomada de [73]):

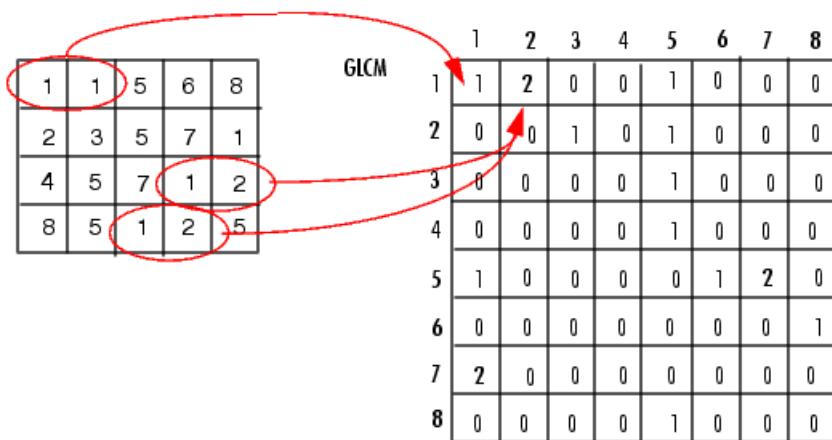


Figura 3.15: Ejemplo de cómo se obtiene un “GLCM” a partir de una matriz de grises obtenida con *graycomatrix*.

Como se puede observar, como el par (1,1) solo se repite una vez, se pone un 1 en la posición (1,1) de la matriz GLCM resultante, sin embargo, el par (1,2) se repite 2 veces, por lo que se pone un 2 en la posición (1,2) del GLCM final.

Cabe añadir que para el cálculo tanto de la dimensión fractal como de la entropía, se necesitaba poder leer la imagen de los gráficos, por lo que se guarda, se lee y se borra al instante del archivo en el que esté ejecutándose el programa. Se hace uso de `imread` [74] para leerla y de `img_as_ubyte(color.rgb2gray(color.rgb2rgb(img)))` [75] para pasarl a blanco y negro totalmente (esto realmente pasa la imagen de un *array* 3D a un *array* 2D).

Una vez calculado esto, faltan las medidas clásicas de competitividad, que son: la desviación típica ( $\sigma$ ) y el índice de balance competitivo de Herfindahl-Hirschman (*HICB*). Ambas fórmulas (ecuaciones 3.3 y 3.4) tienen una traducción sencilla a lenguaje informático. La desviación típica ( $\sigma$ ) queda tal que:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N \left( w_i - \frac{1}{2} \right)^2} \rightarrow \text{math.sqrt(sum((w\_i - 0,5) ** 2 for w\_i in allW\_i)) / len(allW\_i))} \quad (3.3)$$

y el índice de balance competitivo de Herfindahl-Hirschman (*HICB*) tal que:

$$HICB = 100N \sum_{i=1}^N s_i^2 \rightarrow 100 * \text{len(allS\_i)} * \text{sum((s\_i) ** 2 for s\_i in allS\_i)}. \quad (3.4)$$

Tanto “*allW\_i*” como “*allS\_i*” se calcularán por equipo, es decir, por filas, donde la  $i$  corresponde al equipo en cuestión. Ambas se calcularán según sus definiciones (fórmulas 3.5 y 3.6) dadas por el artículo original [1]. Con  $w_i$  definido por:

$$w_i = \frac{\text{victorias del equipo } i \text{ en la liga esa temporada}}{\text{partidos totales del equipo } i \text{ en la liga esa temporada}}, \quad (3.5)$$

donde si la diferencia de puntos entre una jornada y la siguiente es de 3, entonces se le suma una victoria al total de ese equipo y al final se divide por el número de jornadas que corresponde al *largo de la fila* – 2 (menos 2 por ser un valor el nombre del equipo y otro el de la jornada 0). Y  $s_i$  vendrá dado por:

$$s_i = \frac{\text{puntos del equipo } i \text{ en la liga esa temporada}}{\text{suma de puntos de todos los equipos en la liga esa temporada}}, \quad (3.6)$$

donde es calculado con los puntos al final de temporada, y no los ganados (esto solo es relevante para equipos que empiezan la liga con puntos negativos y no con 0, como el Chievo

Verona en la temporada 2018/2019 [60]). Cabe añadir que a  $s_i$  se le dio otra definición al principio por confusión en la traducción de lo escrito en el artículo original [1] y otros 2 diferentes ([14] y [76]), obteniéndose con esta interpretación errónea valores muy distintos a los de estos artículos. Esta definición se verá en la sección 3.4.3 y los resultados obtenidos se verán en la sección 3.4.2.

Por último, una vez se tienen todas las medidas de cada una de las 7 ligas analizadas, falta únicamente el análisis del coeficiente de correlación de Pearson. Para ello se hace uso de la función *corrcoef* [77] de *numpy* que calcula este coeficiente entre las variables que haya en las filas (6 filas, una por medida), apoyándose de los distintos valores que tiene en cada columna (7 columnas, una por liga). Los resultados obtenidos se pasarán a valores absolutos [78] haciendo uso del método *abs* [79], pues es irrelevante que la correlación sea negativa o positiva en este caso, pues lo que interesa es saber si hay relación entre las variables, no cómo es esta [77].

Con esto termina la implementación de la informatización de los cálculos, pero añadir que como se menciona tanto aquí como en la sección de diseño 3.2.2, parte de la implementación de la visualización de resultados 3.3.3 se realizó en paralelo a esta.

### 3.3.3. Visualización de resultados

En esta sección se describirá el uso de la librería *matplotlib.pyplot* [36] y cómo funcionan muchos de sus métodos. Se describirán también los casos especiales de los gráficos de la dimensión fractal, la entropía y la correlación de Pearson.

Las principales funciones a tener en cuenta son: *matplotlib.pyplot.plot* (o *plt.plot*) [80] y *matplotlib.pyplot.scatter* (o *plt.scatter*) [81]. Para usarlas hay que pasarle una lista de coordenadas  $X$ , otra de coordenadas  $Y$  y, una vez hecho esto, se les puede añadir un color, un estilo y una etiqueta o label. Por ejemplo, suponiendo que se tiene una lista de puntos:  $[(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots]$ , y que si se quiere dibujar una línea con esos puntos, habría que usar la función: *plt.plot([x<sub>1</sub>, x<sub>2</sub>, x<sub>3</sub>, ...], [y<sub>1</sub>, y<sub>2</sub>, y<sub>3</sub>, ...])*; y, de igual manera, la función *plt.scatter* en el caso de querer dibujar puntos. Se pueden utilizar ambas funciones para un gráfico de líneas y puntos como se hará para los gráficos del MDS. Los gráficos resultantes se verán en el capítulo 4.

Sobre estos gráficos, para dejarlos con más detalles, más explicativos y más entendibles para el usuario, también se utilizarán las funciones: *plt.annotate* [82] para colocar el nom-

bre o label de la variable (normalmente el nombre de los equipos de fútbol) en el punto final; *plt.legend(loc = 'best')* [83] para colocar una leyenda que explique lo dibujado por colores y estilos colocada en la mejor localización que encuentre; *plt.xlabel* [84] y *plt.ylabel* [85] para poner nombres a los ejes; *plt.xlim* [86] y *plt.ylim* [87] para establecer donde empiezan los gráficos; *plt.figure* [88], para establecer el tamaño de las imágenes, en este caso se usará *plt.figure(figsize = (12, 10))* en todas; y, por último, *plt.show()* [89], que dibujará el contenido que se ha ido añadiendo en las funciones *plt.plot* y *plt.scatter*, además de todos los detalles extras establecidos con el resto de las funciones que se acaban de mencionar.

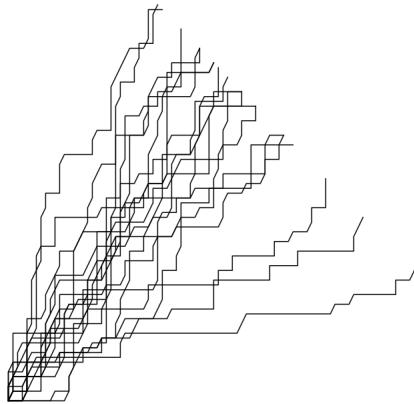
Con todas estas funciones se dibujarán la gran mayoría de los gráficos recorriendo los equipos y, dependiendo de si se necesitan líneas, como en los gráficos del poder complejo, o puntos, como en los gráficos de la clusterización con nubes de puntos, se utilizarán unas funciones u otras. Cabe remarcar el caso de los gráficos de las medidas de competitividad (imagen 4.6), que hará uso de la función *plt.xticks* [90] para rotar los valores en las etiquetas (nombres de las ligas en este caso) en el eje de las *X*.

Continuando con los casos particulares de la entropía y la dimensión fractal, se seguirá lo establecido por el artículo original [1] donde se establece que se debe calcular la dimensión fractal y la entropía tanto de los gráficos del poder complejo como de los gráficos del MDS.

En el caso de la dimensión fractal, esta se debe calcular de los gráficos con solo líneas negras y sin puntos. Y en el caso de la entropía ocurre al contrario, se debe calcular con solo los puntos y sin líneas, asegurándose también de que estos sean negros. Para ello se utilizarán las funciones *plt.plot(x, y, c = 'black')* y *plt.scatter(x, y, c = 'black')*, respectivamente para cada equipo. Por último, en ambos casos se deben borrar de los resultados los ejes, para ello se utiliza la función *plt.axis('off')* [91]. Y, una vez hecho, esto se sigue el proceso explicado en la sección 3.3.2 de guardar la imagen en el repositorio donde se encuentre el código, leerla, borrarla del repositorio y transformarla a blanco y negro (pasar la imagen de un *array* 3D a un *array* 2D).

Un ejemplo de imagen del gráfico de la cual se calcula su dimensión fractal (3.16) es la siguiente:

Poder Complejo:



MDS:

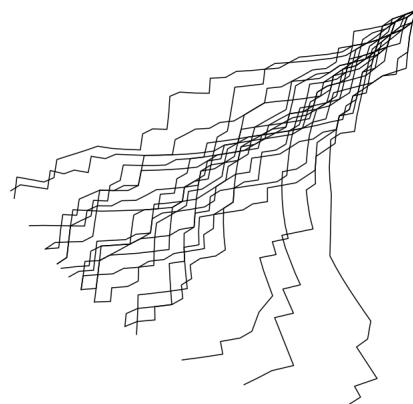


Figura 3.16: Ejemplo de imagen de los gráficos de los que se calcula la dimensión fractal. El resultado obtenido a la izquierda a partir del gráfico del poder complejo y a la derecha a partir del gráfico del MDS. Ambos de “La Liga” (España) durante la temporada 2018/2019.

Como puede verse, tanto para el gráfico del poder complejo como para el del MDS, solo hay líneas negras sin puntos ni ejes.

Y un ejemplo de imagen de un gráfico del que se calcula su entropía (3.17) será el siguiente:

Poder Complejo:



MDS:

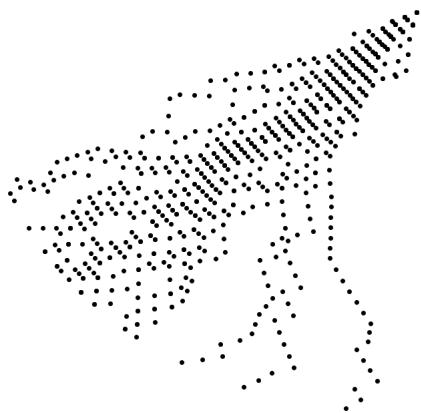


Figura 3.17: Ejemplo de imagen de los gráficos de los que se calcula la entropía. El resultado obtenido a la izquierda a partir del gráfico del poder complejo y a la derecha a partir del gráfico del MDS. Ambos de “La Liga” (España) durante la temporada 2018/2019.

Se observa que en ambos gráficos solo aparecen sus puntos en negro sin líneas entre ellos ni ejes.

También se representan con un gráfico particular los resultados obtenidos por la matriz de correlación de Pearson. En principio se sigue algo parecido a lo hecho en el artículo original [1]. En él se representan las 6 medidas con puntos y se dibujan arcos entre ellas según los valores de esta matriz de correlación, indicando así, cómo de relacionadas están las variables entre si [38]. Para conseguir esto se borran de nuevo los ejes con `plt.axis('off')` [91] y se ponen las medidas en forma de hexágono en función del radio que se desee, en este caso “`radius = -3`”, donde que sea positivo o negativo solo cambia el orden de cómo se colocan las variables (solución de esta implementación tomada de [92]). Una vez colocadas las medidas, se trazarán arcos entre ellas de diferentes colores en función de su valor en la matriz de correlación. En este caso, como la matriz es simétrica, solo se tiene que recorrer la mitad de sus valores.

Cuando ya hayan sido dibujados todos los puntos y los arcos, se quiere poner una leyenda personalizada indicando qué significa cada color, pues si se le asigna una etiqueta a cada arco, esta sale repetida varias veces en la leyenda final. Para ello se hace uso de la función `mpatches.Patch` [93], que asigna una tira de un color a una etiqueta. Esta función junto con el parámetro “`handles`” de `plt.legend` [83] que asignará estas tiras con etiquetas a la leyenda final, terminarán los detalles del gráfico de la matriz de correlación.

Para finalizar, se representará la matriz de correlación con un mapa de calor, asignando de manera parecida al gráfico anterior, unos colores u otros en función de los valores de esta matriz. Para ello se utiliza la función `seaborn.heatmap` [94] a la que se le pasa el `dataframe` (matriz) con los resultados, se le asigna un espectro de colores y, con ello, imprime la matriz como un mapa de calor que es lo que se busca como resultado.

Una vez vista y explicada toda la implementación, habrá que tener cuidado con los posibles errores, pues muchos de estos aparecen en la fase de visualización de datos y fuerzan a volver a fases anteriores.

## 3.4. Pruebas

En último lugar tenemos las pruebas, aquí se verán principalmente los errores. Se verán casos en los que los resultados no fueron los esperados, por lo que se tuvieron que corregir [44].

Algunos de estos errores se dieron en la fase de Informatización de los cálculos, pero no se detectaron hasta la siguiente en la que se visualizaban estos resultados erróneos.

Esto provocó que se tuviera que saltar entre fases y por ello para este ciclo en concreto se seguirá el siguiente orden para las fases: extracción de datos (3.4.1), visualización de resultados (3.4.2) e informatización de cálculos (3.4.3). Se invierte el orden de las dos últimas fases, pues en la sección 3.4.2 se mostrarán los resultados de estos errores y en la siguiente sección 3.4.3 se explicará la implementación de sus soluciones.

### 3.4.1. Extracción de datos

El principal problema que se encuentra en esta fase de extracción de datos es el de la posibilidad de puntuaciones negativas como se mencionaba en la sección 3.3.1 anterior.

Esto se detecta en el Chievo Verona, equipo que quedó en última posición de la liga italiana, durante la temporada 2018/2019 [60]. La sanción con la que comenzó es de  $-3$  puntos, pero han llegado a haber sanciones mayores en la historia del fútbol, por ejemplo la del Luton en la EFL League, equipo que fue sancionado con  $-30$  puntos antes de comenzar su temporada 2008/2009 en dicha liga, o como la del Bolton, que comenzó con  $-12$  en la temporada 2019/2020 de esta liga también [95]. Es por esto que habrá que tener en cuenta a la hora de corregir la función “checkeaPuntos” que los puntos negativos pueden llegar a ser de 2 cifras, es decir, de 3 caracteres contando el signo “ $-$ ”.

El problema destaca a simple vista, pues para el Chievo Verona durante la temporada 2018/2019, se observa el siguiente resultado sin hacer los cambios para admitir puntos negativos (imagen 3.18):

| Serie A/2018-20 | Jornada_00 | Jornada_01 | Jornada_02 | Jornada_03 | Jornada_04 | ... | Jornada_11 | Jornada_12 | Jornada_13 | Jornada_14 | Jornada_15 | Jornada_16 | ... |
|-----------------|------------|------------|------------|------------|------------|-----|------------|------------|------------|------------|------------|------------|-----|
| Chievo Verona   | 3          | 3          | 3          | 2          | 1          |     |            | 1          | 0          | 1          | 2          | 3          |     |

Figura 3.18: Resultados Chievo Verona durante la temporada 2018/2019 con las funciones de “checkeaPuntos” y “checkPuntosIni” sin admitir valores negativos para las puntuaciones de los equipos.

donde como se puede observar, el Chievo Verona obtiene los números descendenteamente, suceso imposible en las ligas de fútbol profesional, y, en el momento señalado por el rectángulo

rojo, pasa a volver a obtenerlos ascendentemente.

La función “checkeaPuntos” tendrá que comprobarse el caso de que los puntos de la jornada anterior sean negativos ( $< 0$ ). En el caso de que lo sean, si las unidades de la jornada actual son menores o iguales a las de la jornada anterior, se cogerán la misma cantidad de caracteres que en la jornada anterior. Es decir, si se pasa de  $-16$  a  $-13$  como  $3 \leq 6$  se sabe que las decenas no han cambiado, por lo que el resultado será de  $-13$ . Esto tiene un caso particular únicamente, que es cuando se pasa de  $-2$  a  $1$ , pues la cifra de las unidades de la jornada actual es menor o igual que la cifra de las unidades de la jornada anterior ( $1 \leq 2$ ), pero en este caso se necesita coger un carácter menos del *string* con los datos brutos (el carácter correspondiente al signo negativo que ahora no tiene). Este caso se tratará en solitario mirando si el penúltimo carácter de la cadena de datos es un “-”, si lo es se cogerán 2 caracteres (caso de que sea  $-2$  o  $-1$  el resultado), y sino solo 1 (caso de que sea  $0$  o  $1$  el resultado).

De manera similar a con las puntuaciones positivas, si las unidades de la jornada actual son mayores a las de la jornada anterior, se tendrá que comprobar si hay que cogerse menos caracteres o no. En este caso, en vez de comprobarse si la puntuación anterior es  $\geq 7$  o  $\geq 97$ , ahora se verificará si esta puntuación anterior es  $\geq -12$  o  $\geq -102$ , pues si se pasa de  $-22$  a  $-19$ , se siguen teniendo que coger 3 caracteres (las 2 cifras y el “-”), sin embargo, si se pasa de  $-12$  a  $-9$ , se tienen que extraer solo 2 caracteres (la cifra de las unidades y el “-”).

En esta ocasión, para conocer si las puntuaciones de las jornadas anteriores se tienen que comprobar con  $-12$  o  $-102$  (o incluso  $-1002$ ), se utiliza la siguiente fórmula (3.7):

$$\text{Comprobador de decenas negativos} = -1 \cdot 10^{\text{caracteres puntuación jornada anterior}} - 2. \quad (3.7)$$

De acuerdo con la misma, si la puntuación de la jornada anterior es tanto  $-11$  como  $-37$  (tienen longitud 3, es decir, 3 caracteres) con el número que se tiene que comparar es con  $-12$ , pues si es  $\geq -12$  significa que hay que cogerse un carácter menos en la jornada actual.

El caso particular de pasar de  $-1$  a  $2$  no se tendrá que hacer por separado, pues si se analiza (fórmula 3.8) se concluye que no hace falta como se ve a continuación:

$$\begin{aligned}
 & \text{Comprobador de decenas negativos} = -1 \cdot 10^{2-2} - 2; \\
 & \text{Comprobador de decenas negativos} = -1 - 2; \\
 & \text{Comprobador de decenas negativos} = -3; \\
 & \quad -1 \geq -3; \\
 & \rightarrow \text{se coge un carácter menos que antes} = \text{se coge solo 1 carácter}; \\
 & \quad \rightarrow \text{resultado} = 2.
 \end{aligned} \tag{3.8}$$

Una vez corregida la función “checkeaPuntos”, habrá que corregir el caso base para que acepte números negativos.

Para el caso base, es decir, para la jornada 1, será necesario un método para comprobar si los datos de esta jornada son negativos o no, pues no vale con únicamente obtenerse el carácter de la derecha del todo como hacía “checkPuntosIni” anteriormente.

El mejor método para esto es complejo, pero consiste en contar el número de elementos que hay entre la última instancia de “:” y la última instancia de “-” en la cadena de los datos recibidos.

De primeras se sabe que si no hay “-” en la cadena de la jornada 1, entonces no es un número negativo. Cabe recalcar, que si fuera la jornada 2 quizás sí, pues la diferencia de goles podría ser negativa con un resultado positivo. Por esto mismo, si hay 2 signos de “-” a la derecha de la última instancia de “:” (a la derecha de aquí concretamente porque sino equipos como el “Paris Saint-Germain” con “-” en su nombre darían problemas si pierden su primer partido), entonces también se puede asegurar que es negativo ese número, por lo que se cogerán todos los caracteres a la derecha del último “-” con este “-” incluido y se transformarán de un *string* a un *integer* con la función *parseInt* [96].

El problema principal ocurre cuando un equipo con puntos negativos gana su primer partido, pues será difícil de diferenciar de un equipo con puntos negativos que haya perdido su primer partido. De forma ilustrada, un ejemplo de ambos casos sería tal que así (Imagen 3.19):

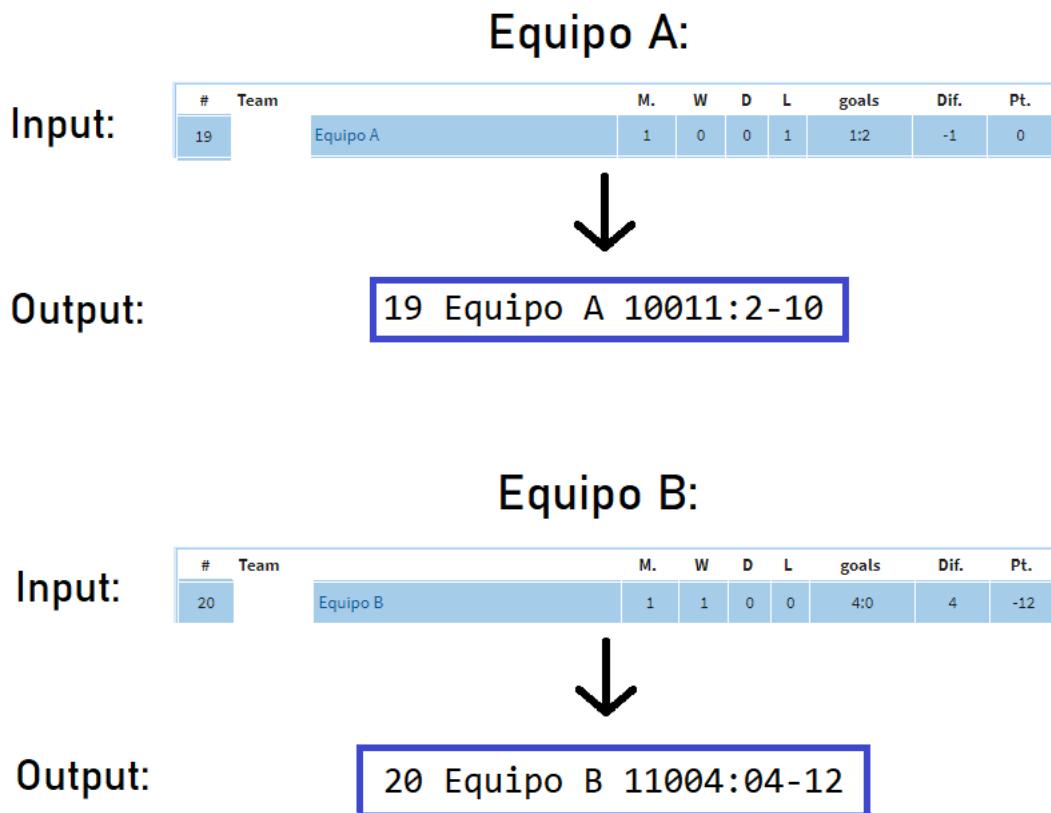


Figura 3.19: Equipo A, pierde su primer partido (1 a 2 en este ejemplo) y pasa de 0 puntos a 0 puntos. Equipo B, gana su primer partido (4 a 0 en este ejemplo) y pasa de -15 puntos a -12.

En la figura puede observar que el Equipo A, como pierde su primer partido 1 a 2 (1 a 2 en este ejemplo, podría ser cualquier otro resultado que le diera la derrota), se queda con los 0 puntos que tenía antes de comenzar el partido (en la jornada 0); y el Equipo B pasa de tener -15 puntos a tener -12 porque gana su primer partido 4 a 0 (4 a 0 en este ejemplo, podría ser cualquier otro resultado que le diera la victoria o el empate).

Siendo el caso de una victoria o empate en el primer partido el peor caso posible, puesto que si pierde, la cadena de datos tiene 2 símbolos “-” a la derecha del último “:”, y entonces se sabría directamente que los puntos son negativos. Observando la imagen 3.19, se observa que la forma de comprobar si es negativo el resultado o no es mirando la cantidad de caracteres que hay entre la última instancia de “:” y la última instancia de “-”, puesto que si es mayor que 1 ( $> 1$ ), la puntuación es negativa y se cogerán todos los caracteres a la derecha del guión con este incluido, pero si es menor o igual a 1 ( $\leq 1$ ), entonces será positivo y se cogerá únicamente el carácter de más a la derecha. Anotar que no vale con comprobar si el carácter de la derecha

del todo es 0, pues se puede pasar de  $-13$  a  $-10$  en la primera jornada perfectamente.

No solo todo esto, sino que como los datos de la jornada 0 no vienen dados por [www.worldfootball.net](http://www.worldfootball.net) [18], no valdrá con únicamente poner un 0, se tendrán que hacer algunos cálculos por el caso de que sean negativos los puntos antes de comenzar la liga. Para ello se obtienen los 4 caracteres que están a la izquierda de los “ $:$ ” y se desecha el de la derecha del todo de estos 4. De esta forma se observa que la primera cifra indica las victorias, la segunda los empates y la tercera las derrotas. Así que para calcularse los puntos de la jornada 0 con la puntuación de la jornada 1 se deduce que si se tiene “100” significa que el equipo en cuestión ganó su primer partido, por lo que se le quitan 3 puntos; “010” significa que empató este partido, por lo que se le quita solo 1 punto; y “001” significa que lo perdió, así que se deja como estaba. Se le quitan puntos y no se le añaden porque se quiere calcular la jornada anterior, no la posterior. Añadir que este método no funcionará en las jornadas posteriores porque se marcarán más goles y no se sabrán cuantos caracteres a la izquierda de los “ $:$ ” hay que cogerse por la posibilidad de que se marquen 10 o más goles.

Cabe añadir que *checkPuntosIni* tiene una debilidad debido a que el número de goles está por entre medias del *string*. La debilidad en cuestión aparece en el caso de que el último equipo marque o le marquen 10 goles en la primera jornada de liga, pero por suerte o por desgracia, este caso no ha ocurrido aún. Añadir que es bastante raro que se de esto, pues en muy pocas ocasiones un equipo profesional ha anotado 10 goles a otro equipo de su misma liga, de hecho, en los 92 años de historia de la liga española (1929 hasta 2021 [97]), solo en 5 ocasiones se han dado resultados en los que un equipo ha marcado 10 goles o más que su rival [98].

Una vez corregido todo esto, como se menciona en la sección 3.3.1, como para obtener los nombres de los equipos se extraen todos los “ $-$ ”, “ $:$ ” y números del *string* de los datos en bruto, en ciertas ocasiones se perderá algún detalle en equipos como el “Schalke 04”, que quedará como “Schalke” solamente.

### 3.4.2. Visualización de resultados

Como se menciona al principio de esta sección 3.4, para las pruebas se cambia el orden seguido en los otros ciclos y se hablará primero de la visualización de resultados y luego de la informatización de cálculos, pues no fue hasta que se observaron los gráficos con errores, que se vieron los fallos en los cálculos. De esta manera aquí se mostrarán las imágenes y resultados

con errores y en la sección de pruebas de Implementación de los cálculos 3.4.3 se describirán los cambios hechos en los cálculos e implementación para corregirlos.

En primer lugar, el error más grande se encontraba en los gráficos del MDS, pues los resultados originalmente eran tal que así (imagen 3.20):

Figura 3.20: Resultados originales del MDS en todas las ligas.

Se observa que hay muchos puntos donde parece que la jornada se va al lado opuesto del gráfico. En concreto, si uno se fija por ejemplo en los puntos naranjas del “Atlético de Madrid”

54

en el primer gráfico, el de “La Liga”, se ven claramente muchos de estos puntos mal colocados.

Para corregir esto se investigaron varias formas de solucionarlo, entre ellas, trazar una línea recta que actuase como eje de simetría y así todos los que estuvieran mal colocados se pondrían al otro lado de este eje. El problema principal de esto, es que el resultado seguía sin ser el esperado como se ve a continuación (imagen 3.21):

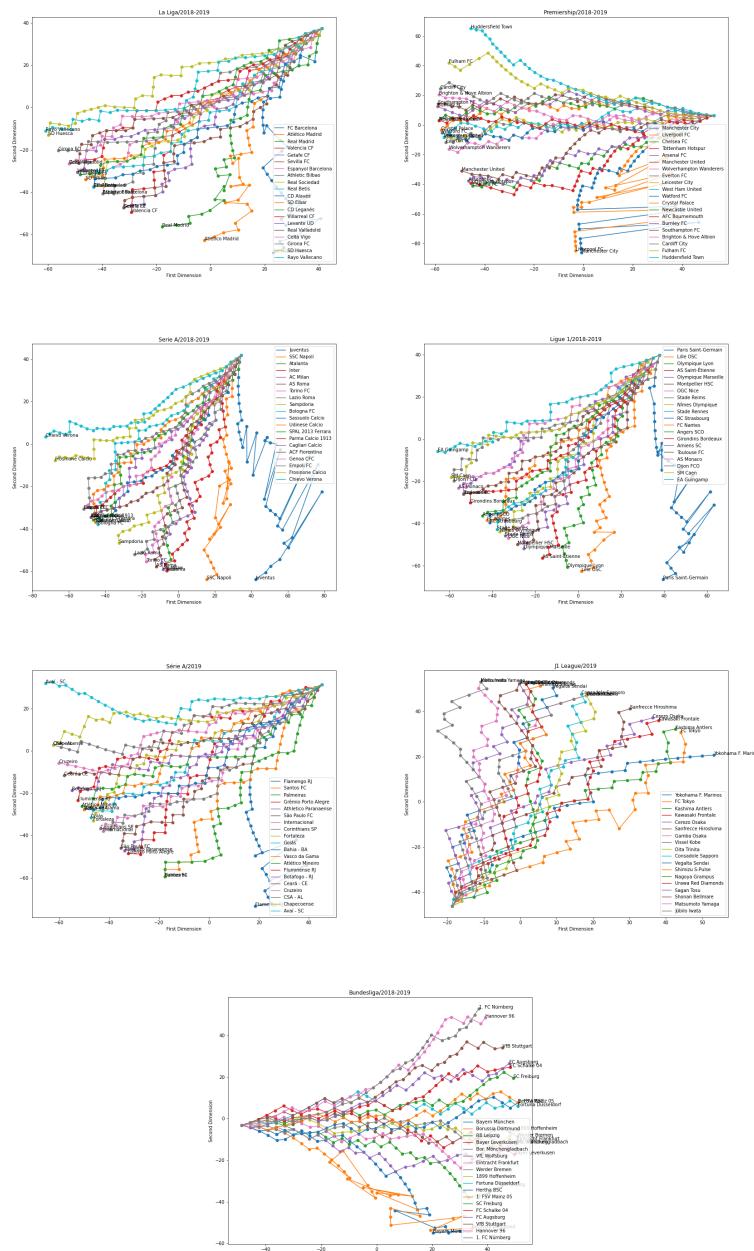


Figura 3.21: Resultados del MDS tras intentar corregirlos con ejes de simetría (función “`fix_MDS_ejeSim`”) en todas las ligas.

Esta vez en la figura se puede observar que los punto del “Atlético de Madrid” de “La Liga” (primer gráfico) están mejor posicionados, pero completamente alejados del lugar correcto. Por este motivo se desechó la idea de que fueran los puntos opuestos a este eje de simetría.

La solución final aplicada consiste en eliminar los puntos mal colocados, y poner esa cantidad de puntos en la mitad del tramo entre los bien colocados. Un ejemplo del resultado con este arreglo es el siguiente (imagen 3.22):

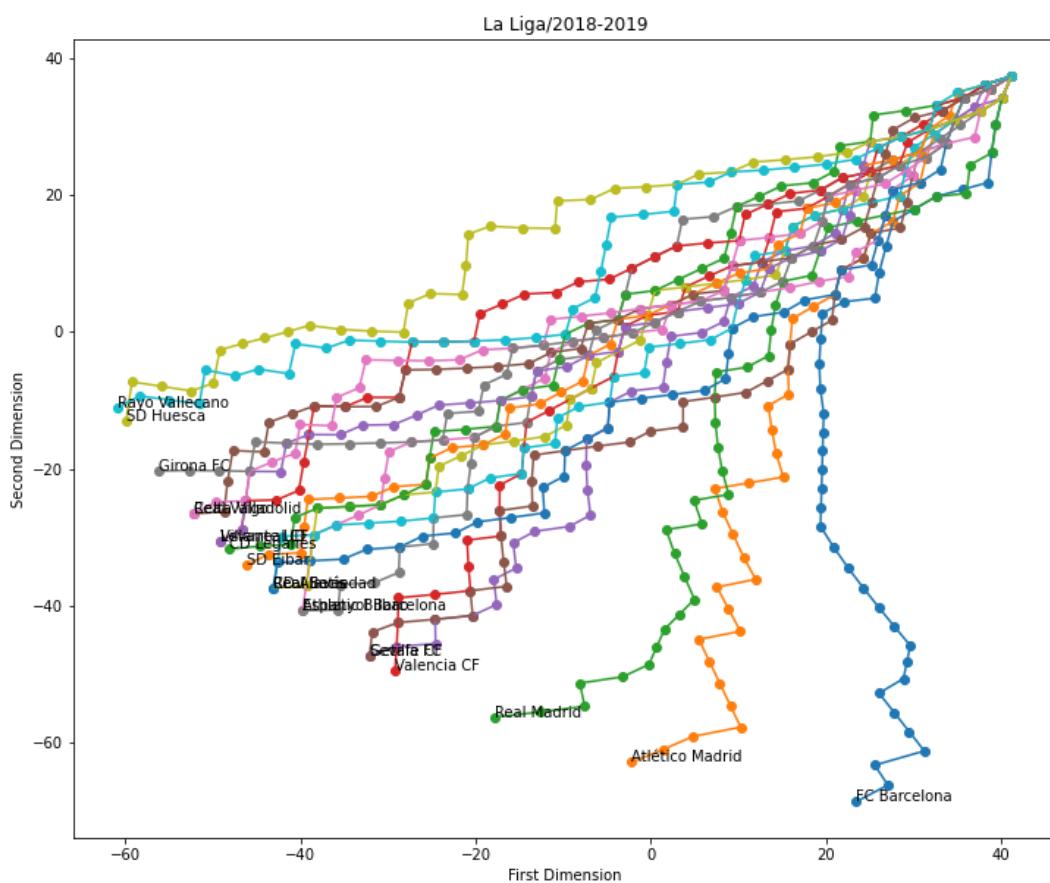


Figura 3.22: Resultados con corrección del MDS con el uso de puntos intermedios (función “*fix\_MDS\_intermedios*”) de “La Liga” (España).

Ahora el “Atlético de Madrid” y el resto que tenían varios puntos mal posicionados, pasan a tenerlos corregidos con una línea entre los últimos 2 puntos bien colocados y la cantidad de puntos borrados equidistantes los unos de los otros.

Esto causará problemas adicionales como qué hacer con la primera y la última iteración o cómo hacer este conteo. Todo ello se resolverá en la sección 3.4.3.

Por último, como se menciona en la sección 3.3.2, el cálculo del índice de balance competitivo de Herfindahl-Hirschman (*HICB*) se obtenía con una fórmula errónea, dando como resultado algo muy alejado de lo esperado (imagen 3.23):

**HICB bien:**

| <b>HICB</b>           |            |
|-----------------------|------------|
| J1 League/2019        | 106.302500 |
| La Liga/2018-2019     | 106.928080 |
| Ligue 1/2018-2019     | 109.307192 |
| Série A/2019          | 109.740238 |
| Serie A/2018-2019     | 111.651944 |
| Bundesliga/2018-2019  | 112.395224 |
| Premiership/2018-2019 | 114.674897 |

**HICB mal:**

| <b>HICB</b>           |               |
|-----------------------|---------------|
| J1 League/2019        | 79170.031217  |
| La Liga/2018-2019     | 110781.250000 |
| Ligue 1/2018-2019     | 159072.702332 |
| Bundesliga/2018-2019  | 222307.479224 |
| Serie A/2018-2019     | 297280.000000 |
| Série A/2019          | 297880.000000 |
| Premiership/2018-2019 | 511898.437500 |

Figura 3.23: Resultados del *HICB* en cada una de las ligas. A la izquierda con el método correcto y a la derecha con el incorrecto.

Aquí vemos que los resultados ni se acercan al esperado que debe oscilar entre 105 y 115. El resultado ni siquiera tiene correlación alguna con el real, es decir, ni con un multiplicador se podría corregir lo obtenido. Esta falta de correlación se ve claramente en el orden diferente de las ligas entre el resultado esperado y el erróneo. Por ejemplo, la “Série A” (Brasil) y la “Bundesliga” (Alemania) se encuentran completamente descolocados en el gráfico de la derecha en comparación al de la izquierda.

Una vez vistos estos resultados erróneos, se tienen que corregir volviendo al ciclo de implementación. Es por eso que las pruebas de la Informatización de los cálculos (3.4.3), donde se explicará la implementación de las correcciones, van después, ya que de esta manera fue como se desarrolló el software.

### 3.4.3. Informatización de cálculos

Comenzando con el motivo por el que ocurren los errores del MDS observados en la imagen 3.20 no se llega a conocer con certeza. Si se cambian todos los parámetros de la función

`sklearn.manifold.MDS` [42], se observa que el que más hace variar el resultado es el “random\_state”, el cual se encarga de dar unos valores iniciales a la función de minimización. Al darle un valor al “random\_state” se fija que el resultado sea igual para ese valor en concreto, en este caso se utiliza “random\_state=5” para todas las ligas. Se sospecha por esto que quizás la razón de los fallos en los gráficos del MDS se encuentre en el hecho de que la función de minimización del estrés no termina de encontrar la mejor solución posible, pues de encontrarla es probable que diera el resultado como debe. La función al ser de minimización causa que el resultado sea variable y no constante. Un ejemplo concreto de cómo son los resultados del MDS sin corrección alguna es el siguiente (imagen 3.24):

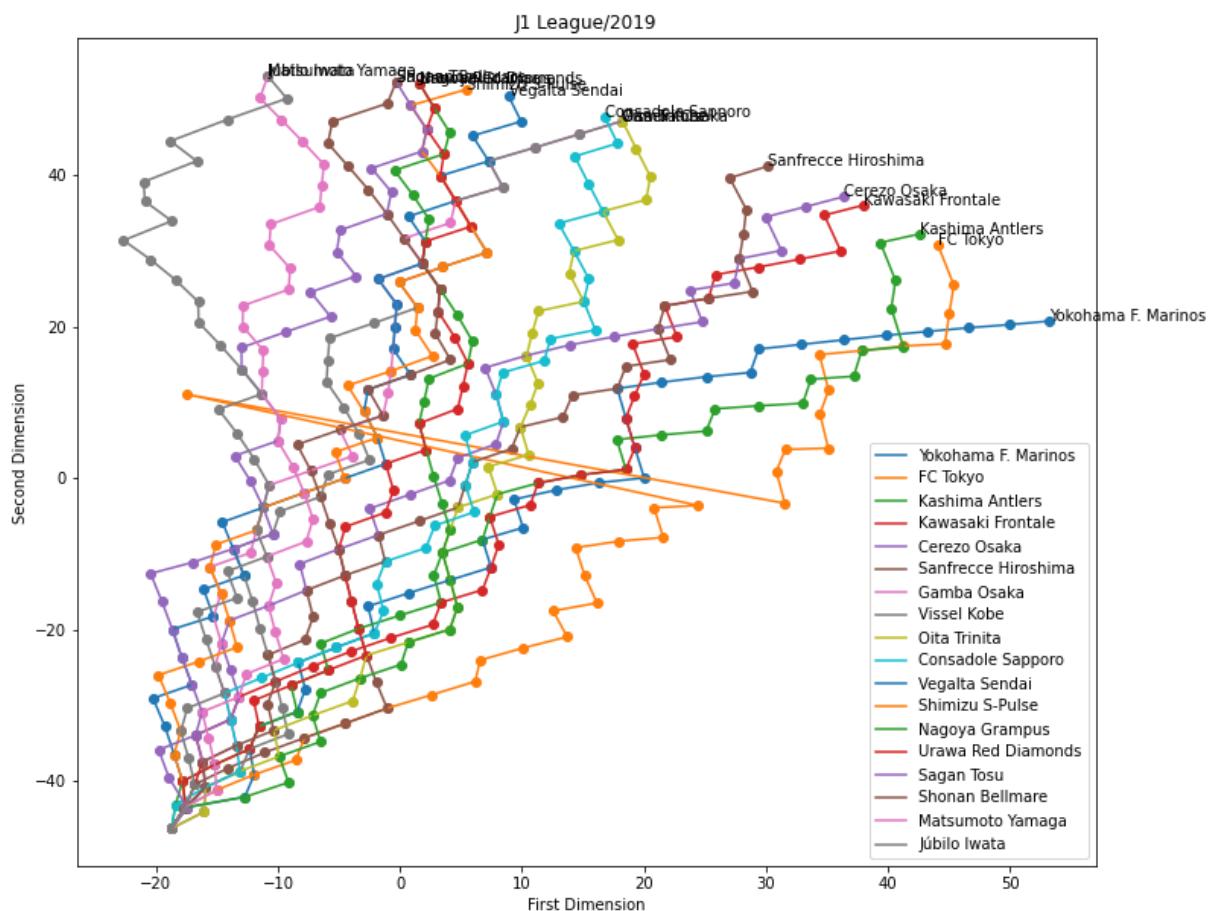


Figura 3.24: Resultados sin corrección del MDS de la “J1 League” (Japón).

Aquí se pueden observar los resultados para el caso concreto de la “J1 League” (Japón). En esta ocasión solo hay un punto mal colocado por lo que será sencillo de seguir. Este es el que se encuentra a mitad de la línea del “FC Tokyo” (naranja).

Antes de comenzar con la implementación, se observa que hay un punto, además del original del que parten todos, que siempre está bien colocado. Este es el último punto del último equipo en la clasificación. Por ello todas las comparativas para comprobar si los puntos están bien colocados o no, se basarán en su posición relativa a este punto.

Volviendo a la corrección de los puntos erróneamente colocados, se opta primero por la función llamada “fix\_MDS\_ejeSim”. Esta lo que hace es trazar un eje desde el punto de partida al (0,0) que es el centro de todos los puntos. Una vez trazado ese eje, se sabe que es posible obtenerse el punto simétrico a otro respecto a un eje, esto se hace con la función llamada “calcPuntoSim” (implementación basada en la referencia [99]). Para comprobar si hay algún error se calcula si la distancia entre un punto y el anterior es mayor a 3 veces la distancia base (distancia entre el primer y el segundo punto), y de serlo, significa que hay alguna irregularidad pues habrá algún punto mal colocado. En tal caso se calculan todos los puntos simétricos para ese equipo y se trazan 2 líneas que serán simétricas, para lo cual se unen los puntos en función de cuál esté más cerca entre el simétrico y el original. Esto se obtiene mediante la función llamada “calcPuntoMasCercano”(implementación basada en la referencia [100]).

Después se comparan las distancias que hay desde el punto final original y desde el punto final simétrico al último punto del último equipo. Se compara con este punto, pues como se ha mencionado anteriormente, es el único que se sabe con certeza que está bien colocado. Y en función de qué punto final esté más cerca, si el original o el simétrico, se cogerá el que corresponda. Si estamos en mitad para arriba de la tabla, se cogerá el que esté más alejado de este punto, pues representa al último equipo de la tabla; y, de igual manera, si está en la mitad baja de la tabla, se cogerá el que tenga el punto final más cercano a este.

En un intento de mejorar los resultados con este método, se prueba a trazar el eje de simetría desde el punto de partida de todos los puntos al punto intermedio que hay entre uno bien colocado y otro mal colocado cuando se produce un salto 3 veces mayor a la distancia base. Con ello el resultado mejora y se obtiene lo que se ve en la imagen 3.21, pero esto no es lo suficientemente bueno. Esto se ve claramente en el resultado del ejemplo anterior de la “J1 League” (Japón) (imagen 3.25):

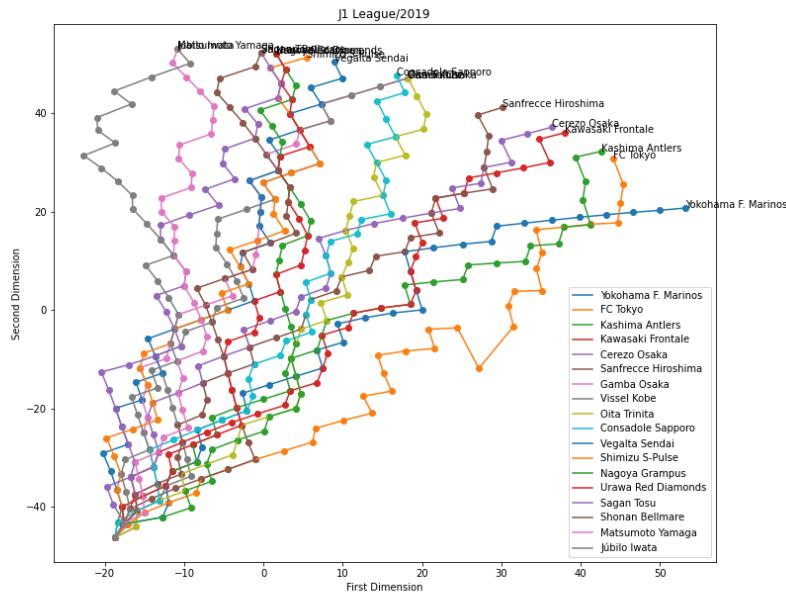


Figura 3.25: Resultados del MDS tras intentar corregirlos con ejes de simetría (función “fix\_MDS\_ejeSim”) de la “J1 League” (Japón).

Se observa claramente que el punto que estaba mal colocado anteriormente en la imagen 3.24, no se encuentra donde debería.

Una vez abordado este método se analizará el mencionado en la sección 3.4.2, que consiste en eliminar los puntos que estén mal colocados y poner esa misma cantidad de puntos en la mitad del tramo entre los bien colocados.

Para ello se desarrolla la función llamada “fix\_MDS\_intermedios”, que comienza observando si el punto final está bien colocado o no, esto se hace aprovechando la implementación de la función “fix\_MDS\_ejeSim” anterior que observa cuál de los puntos está más cerca o lejos del último punto del último equipo (el único que se conoce con certeza que estará bien colocado), si el punto simétrico final o el original. De nuevo, igual que antes, dependiendo de su posición en la tabla se observará si está más cerca o más lejos para determinar si está bien o mal colocado este punto final.

Tras esto, de nuevo se comprueba si la distancia entre un punto con el anterior es mayor o igual a 3 veces la distancia base (distancia entre el primer y el segundo punto). De ser esta distancia así habrá 2 posibilidades, o bien se ha pasado de un punto que estaba bien colocado

a uno que no lo estaba, o bien se ha pasado de un punto que estaba mal colocado a uno que si que estaba bien colocado. Esto se comprueba otra vez haciendo uso del punto simétrico y el original, comparando sus distancias con el punto final de su mismo equipo.

Dado que ya se ha comprobado si el punto va a una posición correcta o a una errónea, se explicará a continuación qué hacer en cada caso. En el caso de pasar de una posición buena a una errónea, se actualizará un índice ( $e$ ) que marcará a partir de qué punto se debe corregir. Este índice se mantendrá estable hasta llegar a pasar de puntos mal colocados a puntos bien colocados, en el caso de que esto ocurra, se verá con un contador cuántas posiciones han habido mal colocadas desde el último bien colocado hasta este, y se trazará una línea entre ambos colocando esa cantidad de puntos equidistantes entre ellos mediante la función “*intermediates*” (implementación basada en la referencia [101]) y borrando todos los puntos mal colocados entre estos 2 bien colocados. Esto también cubrirá el caso de que los primeros puntos estén mal colocados, siempre y cuando se establezca el índice en el primer punto antes de comenzar, pero no el caso de que los últimos puntos estén mal colocados.

Esto permitirá corregir el ejemplo de la “J1 League” (Japón) logrando el siguiente resultado (Imagen 3.26):

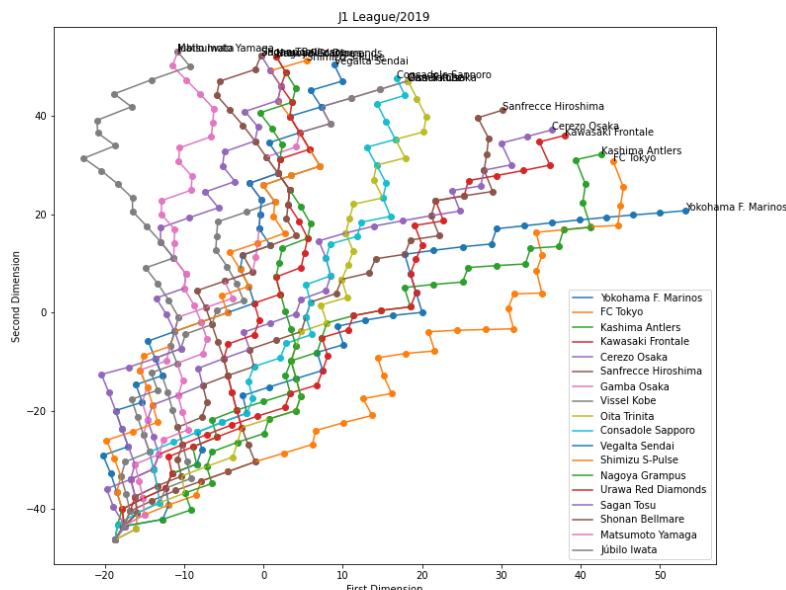


Figura 3.26: Resultados con corrección del MDS con el uso de puntos intermedios (función “*fix\_MDS\_intermedios*”) de la “J1 League” (Japón).

La corrección realizada aún no será suficiente para resolver casos en los que el último punto (o últimos puntos consecutivos) estén mal colocados. Uno de estos casos será el de la “Bundesliga” (Alemania) quedando tal que así (imagen 3.27):

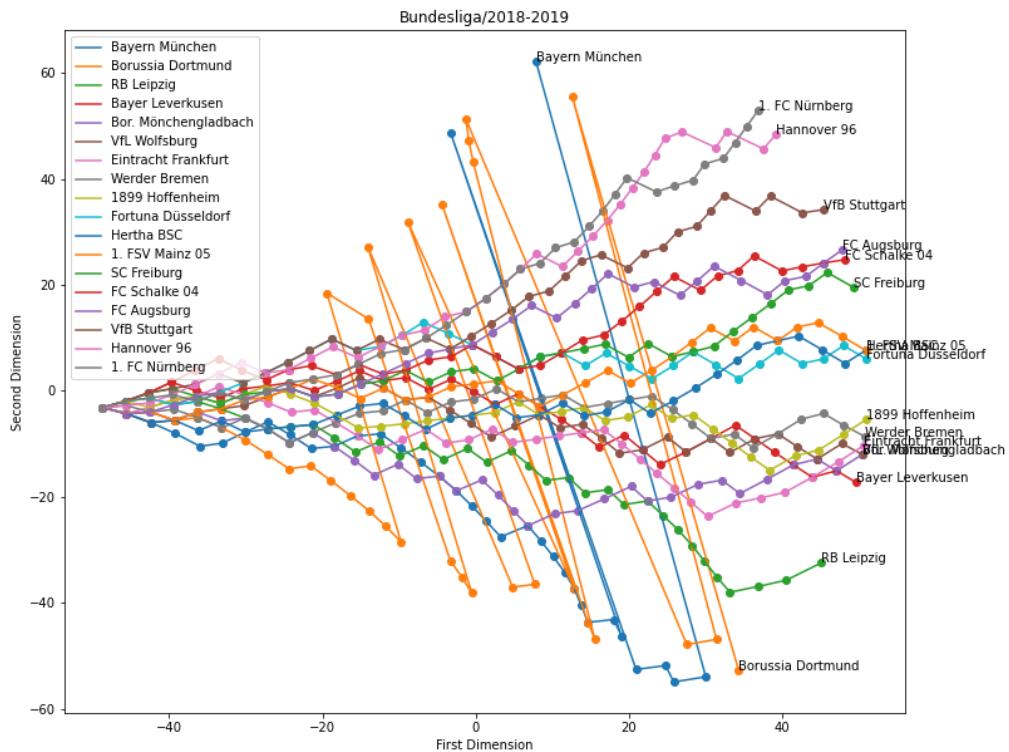


Figura 3.27: Resultados sin corrección del MDS de la “Bundesliga” (Alemania).

Aquí puede verse que el último punto del “Bayern München” (azul) está mal colocado, pues este terminó campeón de su liga con 74 puntos y el equipo que tiene más cerca de este último punto es el “FC Nürnberg” que finalizó en última posición con 19 puntos [102].

Para corregir este caso de que el punto final y los anteriores estén mal colocados se trazará una línea con la dirección del punto de partida al último punto bien colocado en ese equipo. Con esta dirección se dibujarán desde el último bien colocado un total de puntos igual a la cantidad de puntos finales mal colocados. Se hará con una distancia de 2 veces la distancia base, pues se obtiene un resultado mejor que con solo 1 vez la distancia base. Esto se hará haciendo uso de la función llamada “calcPuntoConRectaDistPunto” (implementación basada en la referencia [103]).

Una vez finalizada y aplicada la función “fix\_MDS\_intermedios” sobre los resultados originales, quedarán unos resultados como los que se pueden ver en la imagen 4.4. En concreto para la “Bundesliga” (Alemania) los resultados obtenidos se muestran en la imagen 3.28:

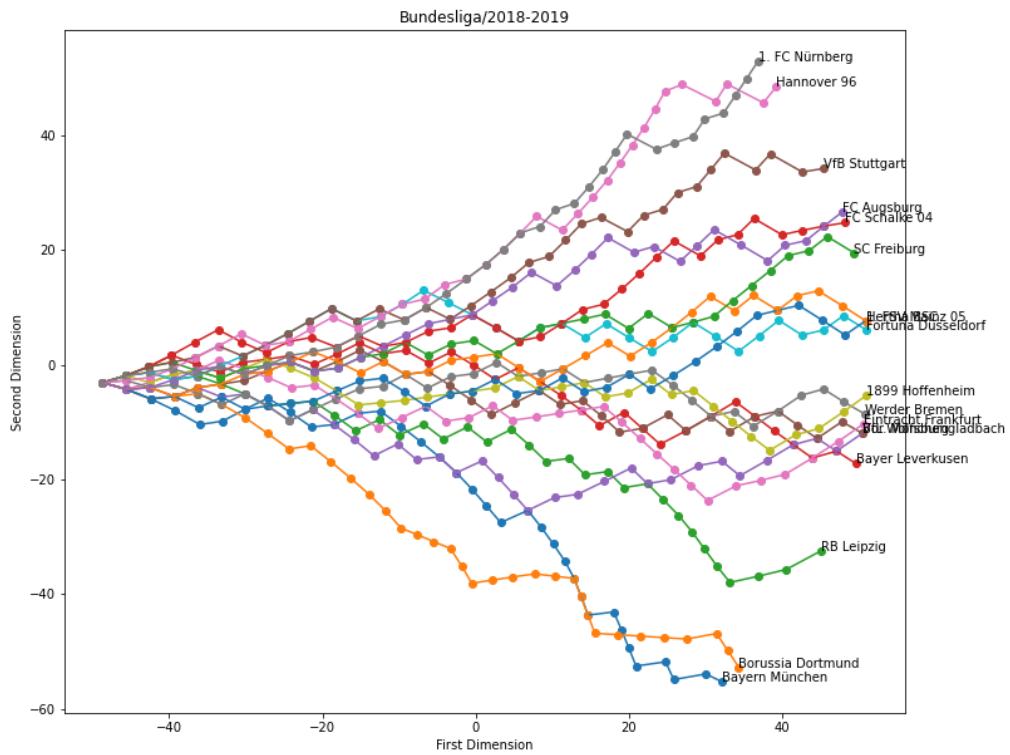


Figura 3.28: Resultados con corrección del MDS con el uso de puntos intermedios (función “*fix\_MDS\_intermedios*”) de la “Bundesliga” (Alemania).

Ahora puede observarse que el punto final del “Bayern München”, que ahora se encuentra abajo a la derecha, queda corregido junto con el resto de puntos mal colocados.

En cuanto a los errores en el cálculo del *HICB* mostrados en la figura 3.23 vienen dados por una mala interpretación de la frase “[...] where  $w_i$  and  $s_i$  denote the proportions of wins and of points scored in a season by the  $i$ th team, respectively.” en el artículo original [1], que viene a significar “[...] donde  $w_i$  y  $s_i$  denotan la proporción de victorias y de puntos anotados en una temporada por el  $i$ ésimo equipo, respectivamente” (se describía de forma similar también en los artículos [14] y [76]). Esto provocó que en vez de aplicarse el  $s_i$  correctamente como se describe en la fórmula 3.6, se acabó usando la siguiente fórmula (3.9):

$$s_i = \frac{\text{puntos del equipo } i \text{ en la liga esa temporada}}{\text{puntos totales que podría haber conseguido el equipo } i \text{ en la liga esa temporada.}} \quad (3.9)$$

Este problema, a priori grande, se soluciona simplemente cambiando de lugar la variable llamada “*totalPuntos*” que irá aumentando cada vez que se observe un aumento en los puntos obtenidos por un equipo mientras se recorre la matriz de datos. Después esta variable será la encargada de dividir los puntos del equipo *i* en la liga esa temporada. El cambio de lugar consiste en colocar *totalPuntos* = 0 antes de entrar en el *for* que se recorre todos los equipos, no después de entrar en ese *for*, pues colocarlo dentro de este *for* significaría tomar la fórmula anterior 3.9 como correcta.

Se observaron algunas pruebas y errores más en los cálculos, pero los 2 aquí mencionados fueron los más relevantes y complicados de resolver.

# Capítulo 4

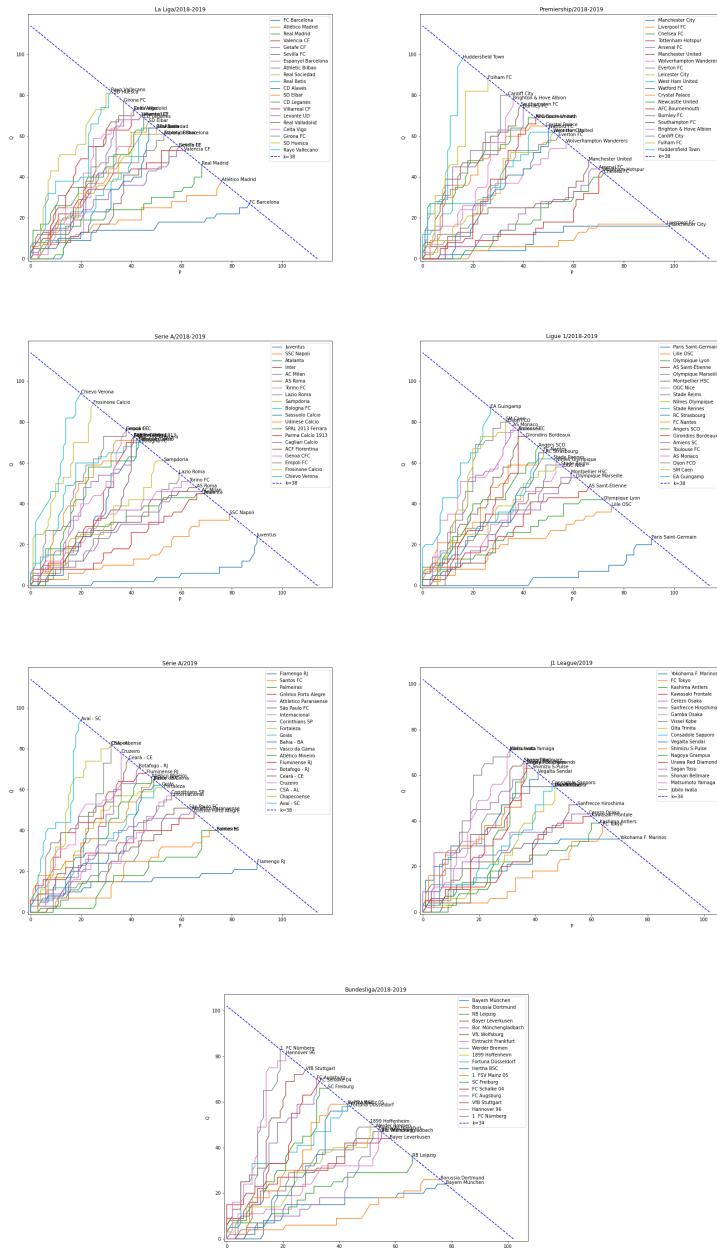
## Resultados obtenidos

Una vez finalizados todos los ciclos del desarrollo de software solo queda mostrar los resultados. Todos los resultados seguirán las pautas establecidas por el artículo: “Modeling and visualizing competitiveness in soccer leagues” [1]. En el capítulo 5 se compararán estos resultados con los de este artículo [1].

No se entrará en profundidad en el análisis de los resultados obtenidos, pues ese es el contenido principal del Trabajo Fin de Grado de Matemáticas [2] junto con su explicación.

Añadir que para todos los gráficos se seguirá el siguiente orden de arriba a la izquierda a abajo a la derecha: “La Liga” (España), “Premiership” (Inglaterra), “Serie A” (Italia), “Ligue 1” (Francia), “Série A” (Brasil), “J1 League” (Japón) y “Bundesliga” (Alemania).

En primer lugar están los gráficos del poder complejo  $S_i(k)$  (imagen 4.1):


 Figura 4.1: Gráficos del poder complejo  $S_i(k)$  de todas las ligas analizadas.

donde los equipos con mejores puntuaciones acaban abajo a la derecha de sus respectivas gráficas, mientras que los equipos con peores resultados acaban arriba a la izquierda. De hecho, como se indica en la implementación (3.3.2), cada victoria te desplaza 3 casillas a la derecha, cada derrota 3 casillas hacia arriba y cada empate 1 casilla a la derecha y 2 hacia arriba. También se puede observar que los resultados parecen indicar que los equipos están divididos por grupos según sus objetivos ya a mitad de temporada incluso.

## CAPÍTULO 4. RESULTADOS OBTENIDOS

---

En segundo lugar se obtienen los dendrogramas resultantes de las posiciones de los puntos finales de los gráficos del poder complejo (imagen 4.2):

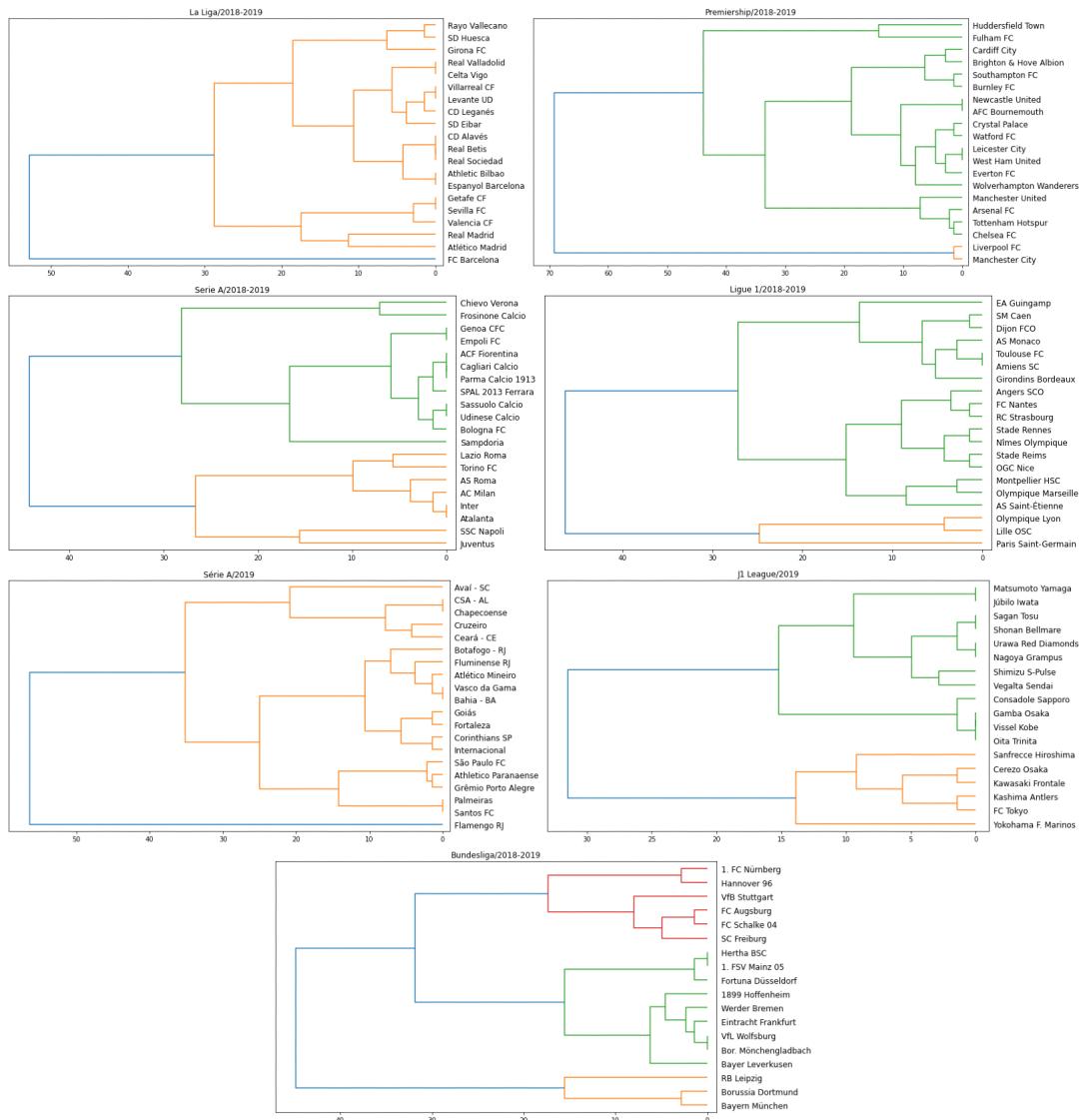


Figura 4.2: Dendrogramas de todas las ligas analizadas.

En estos dendrogramas pueden verse los equipos ordenados de mejor a peor resultado en la “Serie A” (Italia) y la “Ligue 1” (Francia) (tercer y cuarto dendrograma), y de manera opuesta en el resto de ligas. En cuanto a los resultados se observa cómo de cerca están de juntarse unos grupos u otros, por ejemplo, en “La Liga” (España) (primer dendrograma) se puede observar en función de dónde se juntan los grupos o clústeres, que el “FC Barcelona”, el “Atlético de Madrid” y el “Real Madrid” podrían formar un grupo separado al resto de equipos, puesto que se pueden separar con una línea vertical que divida este grupo del resto.

## CAPÍTULO 4. RESULTADOS OBTENIDOS

En este caso la línea vertical se trazará para que siempre divida en 5 grupos o clústeres como se menciona en la sección 3.3.2, quedando los grupos así (imagen 4.3):

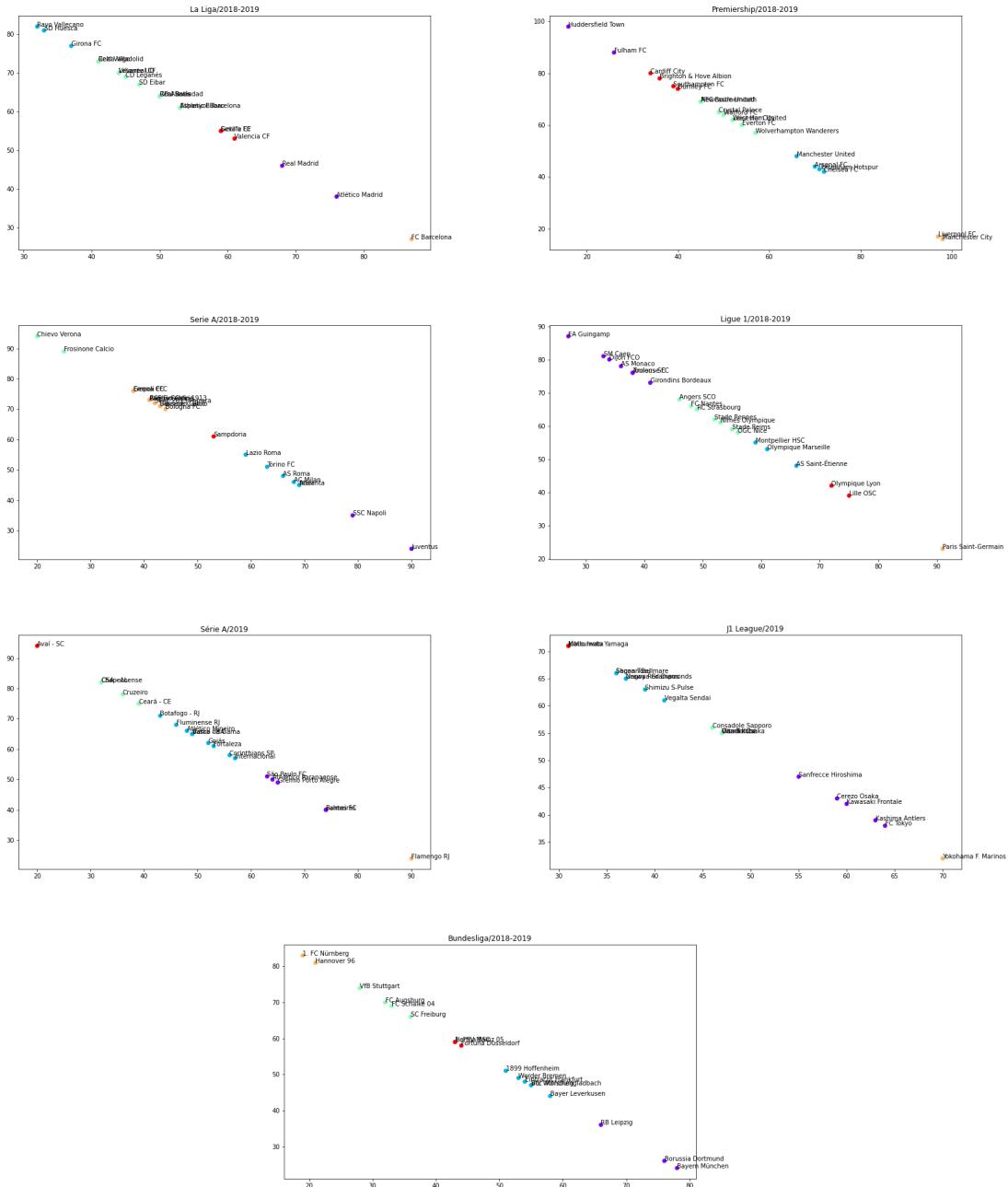


Figura 4.3: Clústeres resultantes a partir de los dendrogramas de todas las ligas analizadas.

En la figura anterior se manifiesta cada uno de los grupos o clústeres resultantes con una nube de puntos de un color diferente. Estos puntos representan los puntos finales de cada equipo en el gráfico del poder complejo, es decir,  $S_i(R)$ , donde  $R$  es el número total de rondas (jornadas).

## CAPÍTULO 4. RESULTADOS OBTENIDOS

A continuación se aplica el proceso de escalamiento multidimensional o MDS y se obtienen los siguientes gráficos (imagen 4.4):

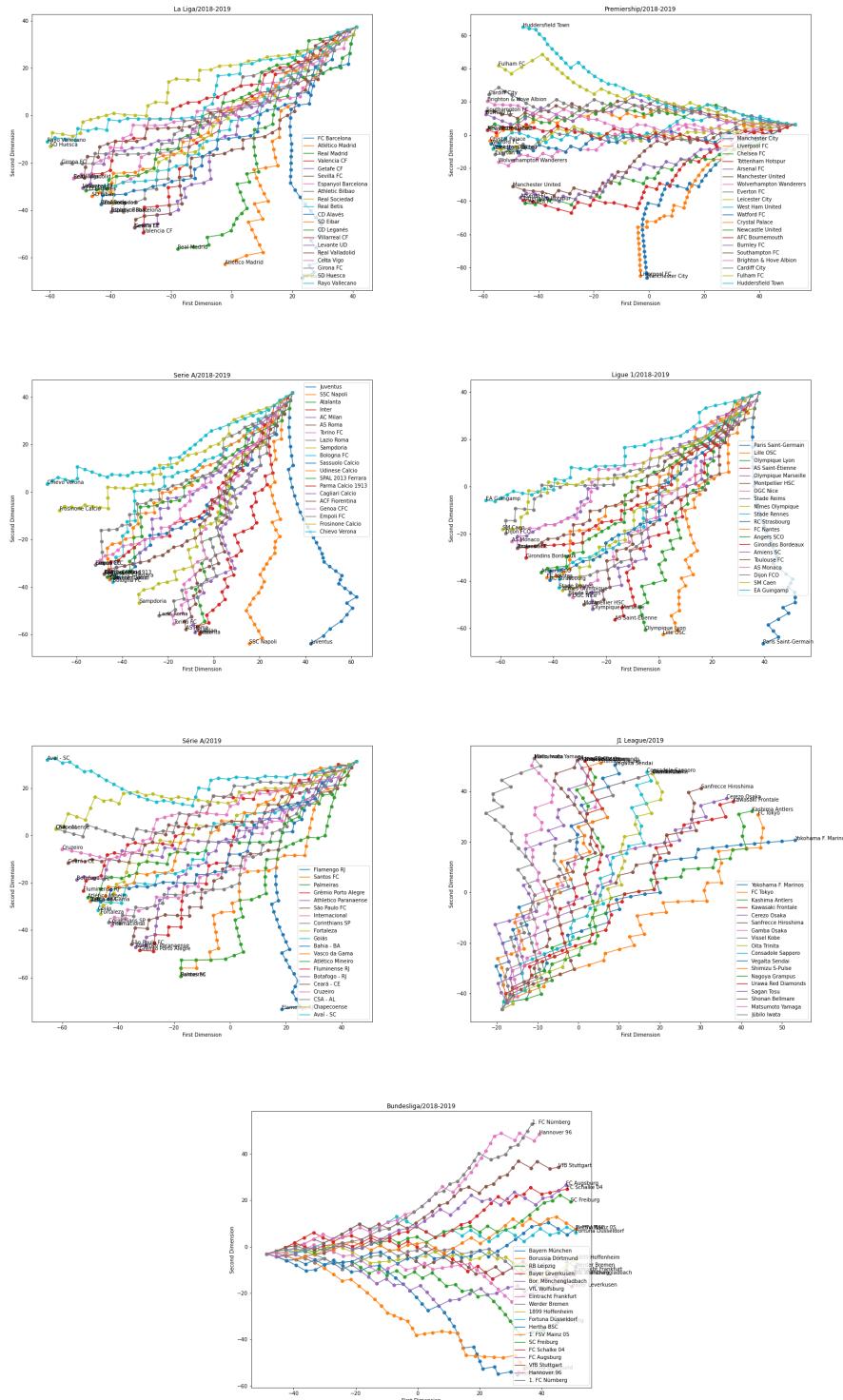


Figura 4.4: Gráficos del MDS de todas las ligas analizadas.

Puede verse que los resultados son similares a los de los gráficos del poder complejo, pero parecen encogidos o extendidos y cambiados de orientación. Esto se debe a que el proceso de escalamiento multidimensional toma los puntos de la tabla del poder complejo, obtiene una matriz de distancias y, con ella, infiere una lista de puntos de 2 dimensiones. Es decir, el gráfico del proceso del MDS tiene que ser muy parecido y casi equivalente al del poder complejo, pues lo que viene a hacer es lo siguiente (4.1):

$$\begin{aligned}
 & \text{Matriz de puntos } (S_i) \rightarrow \text{Lista de puntos } (S_i) \rightarrow \\
 & \rightarrow \text{Matriz de distancias } (S_i) \text{ (dist Manhattan)} \rightarrow \\
 & \rightarrow \text{Escalamiento Multidimensional (MDS)} \rightarrow \\
 & \rightarrow \text{Lista de puntos (MDS)} \rightarrow \text{Matriz de puntos (MDS)}. \tag{4.1}
 \end{aligned}$$

Esto parece poco práctico, pero el motivo principal por el que se hace es porque se quieren conseguir gráficos equivalentes al del poder complejo, pero que sean diferentes a los que ya se tienen.

Una vez se han obtenido tanto los gráficos del poder complejo como los del MDS, se pueden calcular las diferentes medidas de competitividad a partir de ellos, concretamente la dimensión fractal  $b$  y la entropía  $H$  de gráficos como los de las imágenes 3.16 y 3.17, respectivamente.

Con estas medidas obtenidas a partir de los gráficos y las medidas clásicas de competitividad, en concreto la desviación típica  $\sigma$  y el índice de balance competitivo de Herfindahl-Hirschman  $HICB$ , se obtiene la siguiente tabla (4.5):

|                       | Dim fractal Poder Complejo (b_S) | Dim fractal MDS (b_MDS) | Entropía Poder Complejo (H_S) | Entropía MDS (H_MDS) | Desviación típica | HICB       |
|-----------------------|----------------------------------|-------------------------|-------------------------------|----------------------|-------------------|------------|
| J1 League/2019        | 1.473252                         | 1.521259                | 0.351810                      | 0.351637             | 0.169809          | 106.302500 |
| La Liga/2018-2019     | 1.472330                         | 1.467743                | 0.427392                      | 0.421594             | 0.191220          | 106.928080 |
| Ligue 1/2018-2019     | 1.458307                         | 1.447789                | 0.442381                      | 0.430708             | 0.210362          | 109.307192 |
| Série A/2019          | 1.447608                         | 1.477899                | 0.440547                      | 0.437597             | 0.197807          | 109.740238 |
| Serie A/2018-2019     | 1.448044                         | 1.449811                | 0.458688                      | 0.444231             | 0.220174          | 111.651944 |
| Bundesliga/2018-2019  | 1.442888                         | 1.487070                | 0.385077                      | 0.380987             | 0.205766          | 112.395224 |
| Premiership/2018-2019 | 1.432364                         | 1.444388                | 0.457683                      | 0.443594             | 0.208128          | 114.674897 |

Figura 4.5: Tabla con todas las medidas (columnas) de todas las ligas analizadas (filas) ordenadas de menor a mayor  $HICB$ .

En ella se ordena de menor a mayor  $HICB$ , pues es el principal indicador de la competitividad junto con la desviación típica que, al igual que el  $HICB$ , cuanto menor sea su valor, mayor

## CAPÍTULO 4. RESULTADOS OBTENIDOS

---

será la competitividad de esa liga, es decir, son inversamente proporcionales su valor y su competitividad. En contradicción a esto están las medidas de la dimensión fractal y de la entropía, las cuales indican una menor competitividad cuanto menor sea su valor y viceversa, es decir, son directamente proporcionales su valor y su competitividad [1] [104].

Por otra parte, si la desviación típica es baja, indica que casi todos los equipos ganarán la misma cantidad de partidos, mientras que si es alta, implica que hay equipos que ganan siempre y equipos que pierden siempre, dejando así una liga muy poco competitiva. Por lo que cuanto menor sea la desviación típica, más cerca estarán los resultados de la media, así que la liga estará más apretada y mayor será la competitividad.

De manera similar, el *HICB* es una medida que indica cómo de concentrados o dispersos están los puntos en la liga. Si el *HICB* es bajo, implica que los puntos están más concentrados, y si es alto, conlleva que los puntos están más dispersos y repartidos. Así que al igual que a la desviación típica, a menor *HICB*, mayor competitividad.

Mientras, por otro lado, una alta entropía o/y dimensión fractal se asocian con una competitividad alta porque ambas son medidas que miden cómo de dispersa u ordenada está una imagen. Para determinar que la competitividad es alta, lo que se busca es el desorden y el caos en las imágenes que representan los puntos de los equipos (los gráficos del poder complejo y del MDS), puesto que de esta forma no se puede conocer qué equipo acabará en qué posición. Y como a mayor desorden, mayor será la dimensión fractal y la entropía, se puede concluir que cuanto mayores sean la dimensión fractal y la entropía, mayor será la competitividad.

Con esta tabla de medidas (4.5) se obtendrán las siguientes gráficas (Imagen 4.6):

## CAPÍTULO 4. RESULTADOS OBTENIDOS

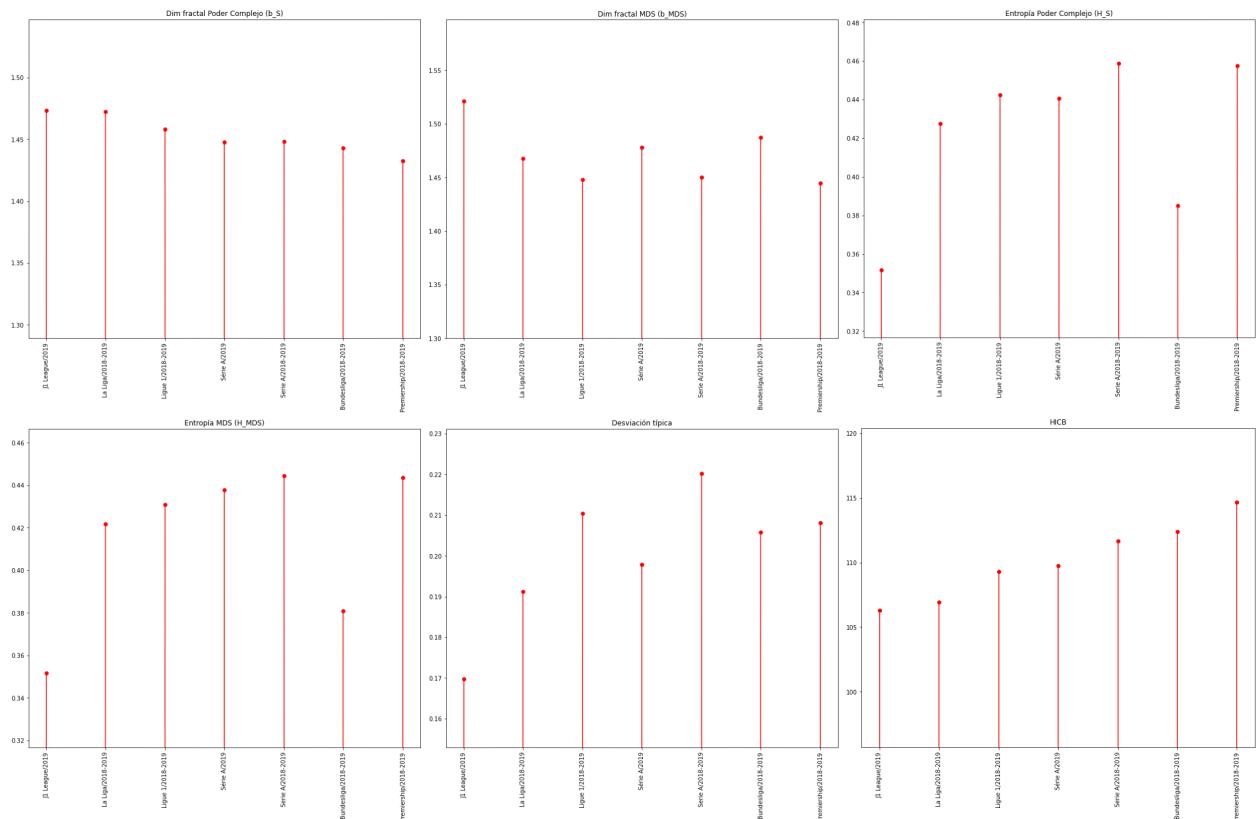


Figura 4.6: Gráficas de cada medida con todas las ligas analizadas.

En ellas se observa que ninguna sigue el mismo orden que el *HICB* o el mismo orden invertido, por lo que será difícil encontrar variables que tengan un alta correlación con esta, ya sea correlación directa o inversa [47].

En cualquier caso, una vez se obtienen todas las medidas de las 7 ligas analizadas, se calcula el coeficiente de correlación de Pearson que tienen entre ellas para comprobar como de relacionadas están las unas con las otras. Como resultado se obtiene la siguiente matriz de correlación [38] (imagen 4.7):

## CAPÍTULO 4. RESULTADOS OBTENIDOS

---

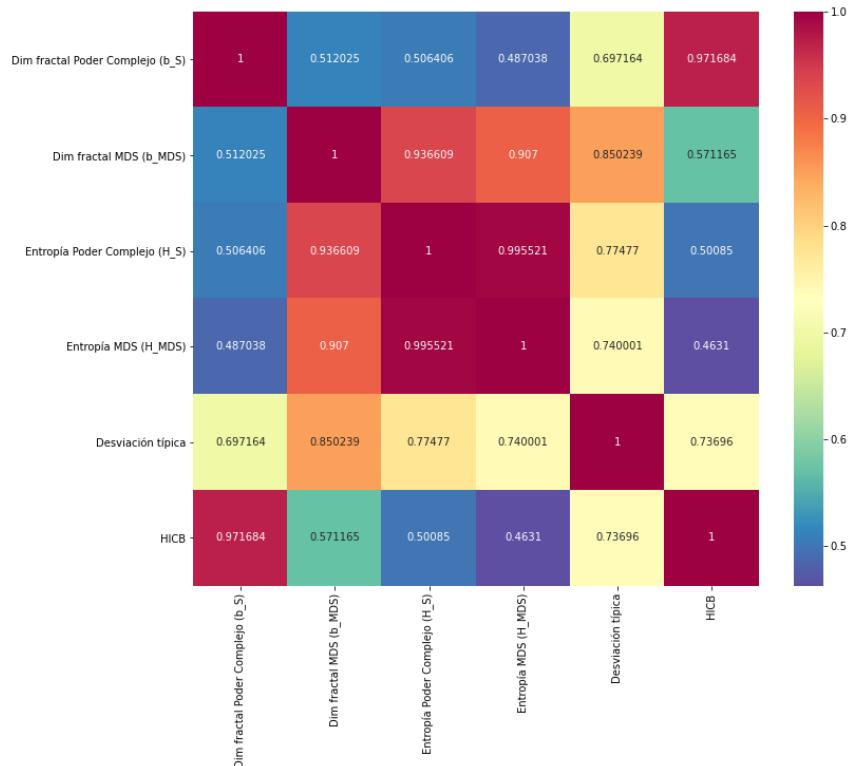


Figura 4.7: Matriz de correlación calculada con el coeficiente de correlación de Pearson entre todas las medidas usando los resultados de todas las ligas analizadas.

De dicha matriz se puede inferir un gráfico más visual (imagen 4.8):

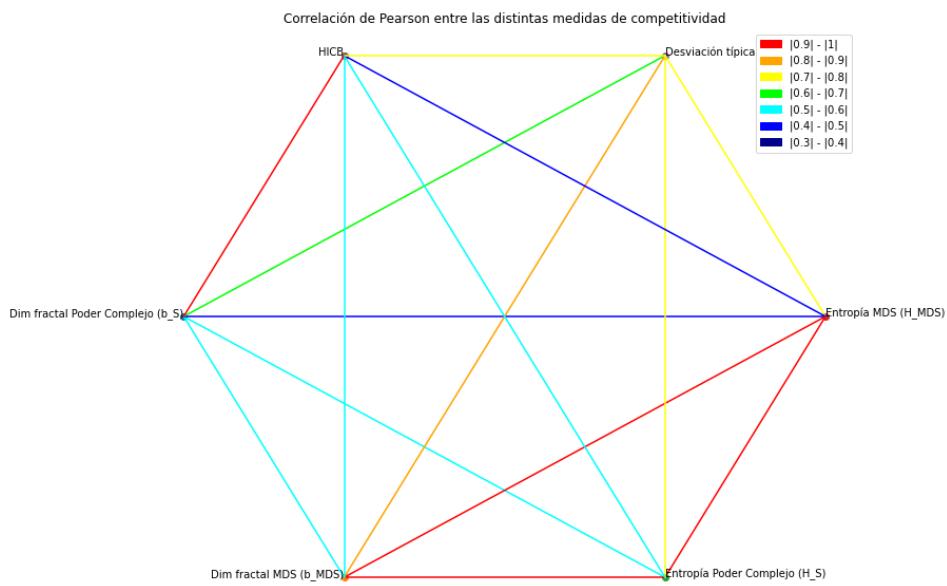


Figura 4.8: Gráfico representativo de la matriz de correlación de las medidas de competitividad de las ligas de fútbol profesional.

Por desgracia en este gráfico se pierde un poco de detalle frente a la matriz de correlación 4.7.

Tanto para matriz de correlación como en su gráfico se utilizan valores absolutos [78], pues, como se menciona en la sección 3.3.2, es irrelevante que la correlación sea negativa o positiva en este caso, pues lo que interesa es saber si hay relación entre las variables, no cómo es esta [47].

Con este último gráfico se cumplen todos los casos de uso que se veían en la sección 3.2.3 en la imagen 3.9.

# Capítulo 5

## Conclusiones

Finalmente, tras haber informatizado de manera exitosa todo el contenido del artículo: “Modeling and visualizing competitiveness in soccer leagues” [1], se procede a comparar sus los resultados con los obtenidos en este TFG.

Los resultados a comparar serán los de los gráficos del poder complejo, los dendrogramas, los gráficos del MDS, la tabla de resultados y la gráfica de la matriz de correlación

Comenzaremos por los gráficos del poder complejo  $S_i(k)$  del artículo (figura 5.1 tomada de la sección 2 de [1]):

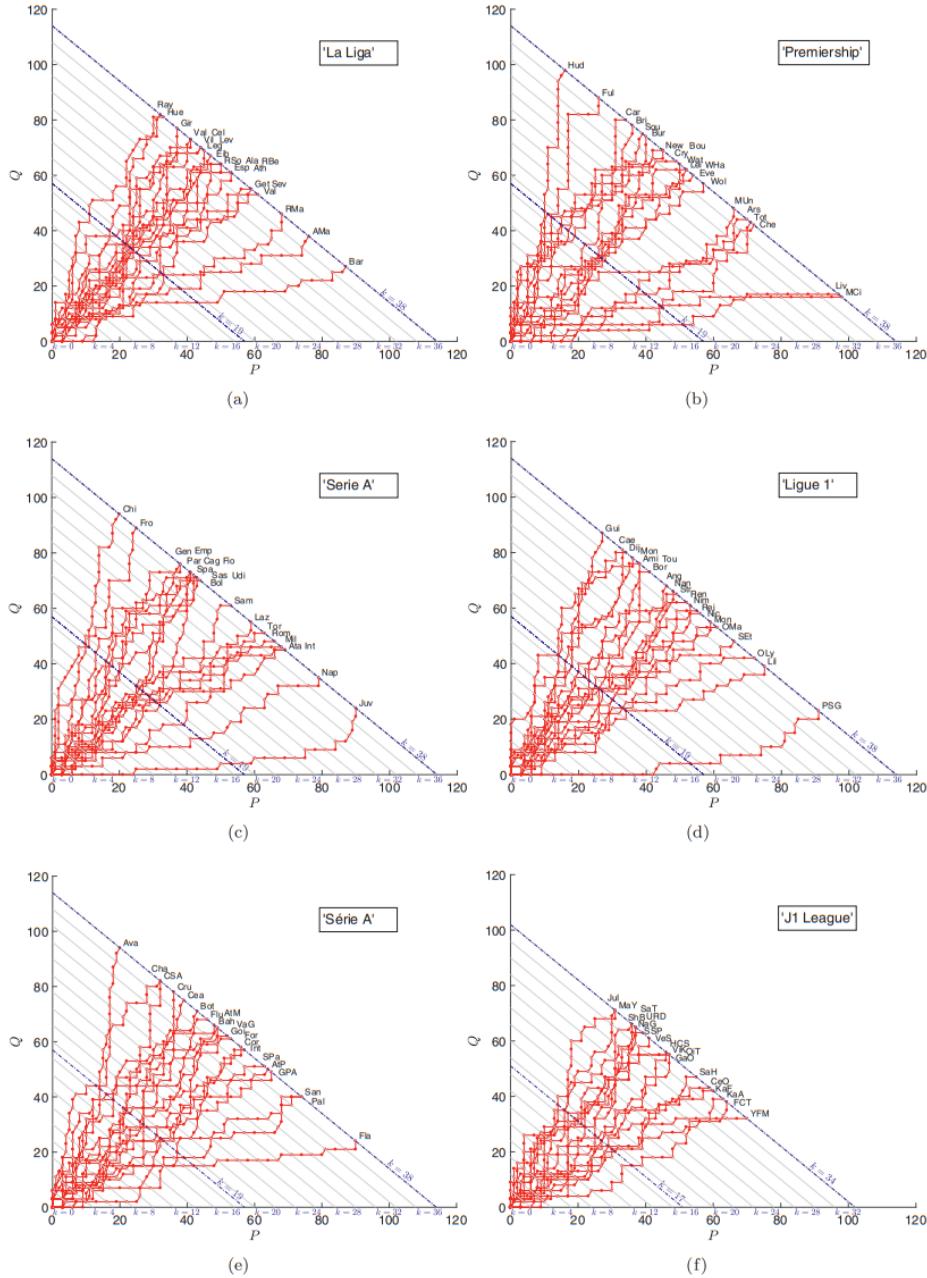


Figura 5.1: Evolución de  $S_i(k)$  frente a  $k$  en 6 ligas: (a) “La Liga”; (b) “Premiership”; (c) “Serie A”; (d) “Ligue 1” (e) “Série A”; (f) “J1 League”. Las líneas diagonales azules representan las ondas para valores sucesivos de  $k$ .

Puede observarse que estos gráficos son prácticamente iguales a los obtenidos en este TFG (imagen 4.1), donde la única diferencia palpable está en que a los gráficos de este TFG se les han añadido colores para dejar más claro qué equipo representa cada línea. Esta similitud se debe principalmente a la ausencia de factores aleatorios, problema que relucirá en los gráficos

del MDS.

El único dendrograma aportado por el artículo [1] es el de “La Liga” (figura 5.2 tomada de la sección 2 de [1]):

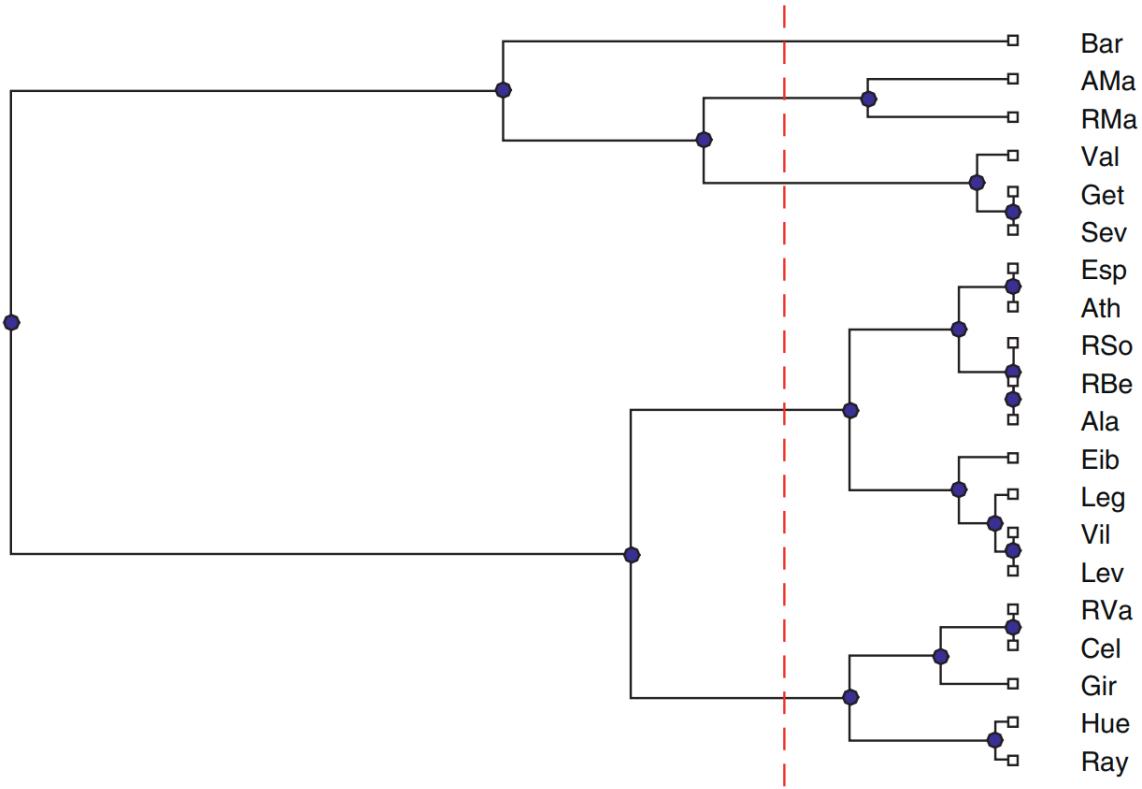


Figura 5.2: Dendrograma generado para “La Liga” al final de temporada ( $S_i(R)$ ) usando el agrupamiento enlace media o promedio [3] como método y la distancia euclidiana como métrica [4]. La línea intermitente vertical de color rojo, secciona el dendrograma para obtener los diferentes clústeres, 5 en este caso.

Como se puede observar, este dendrograma se ve muy diferente al obtenido en el TFG observado en la imagen 4.2, sobre todo si uno atiende a la zona central del dendrograma.

En primer lugar se sospechó que esto fuera por un fallo en la interpretación de los signos “|” descritos en el artículo [1], pues en el caso de que se tomasen como módulo, entonces la distancia tomada entre los puntos sería la euclidiana (“metric=‘euclidean’”), mientras que si se interpreta como valor absoluto, entonces esta distancia sería la de Manhattan (“metric=‘cityblock’”) [4], ambas explicadas en la sección 3.3.2. Por ello se calcula el dendrograma con la distancia de Manhattan o cityblock (imagen 5.3) como nueva métrica y el resultado es prácticamente

idéntico al obtenido con la euclíadiana (imagen 4.2):

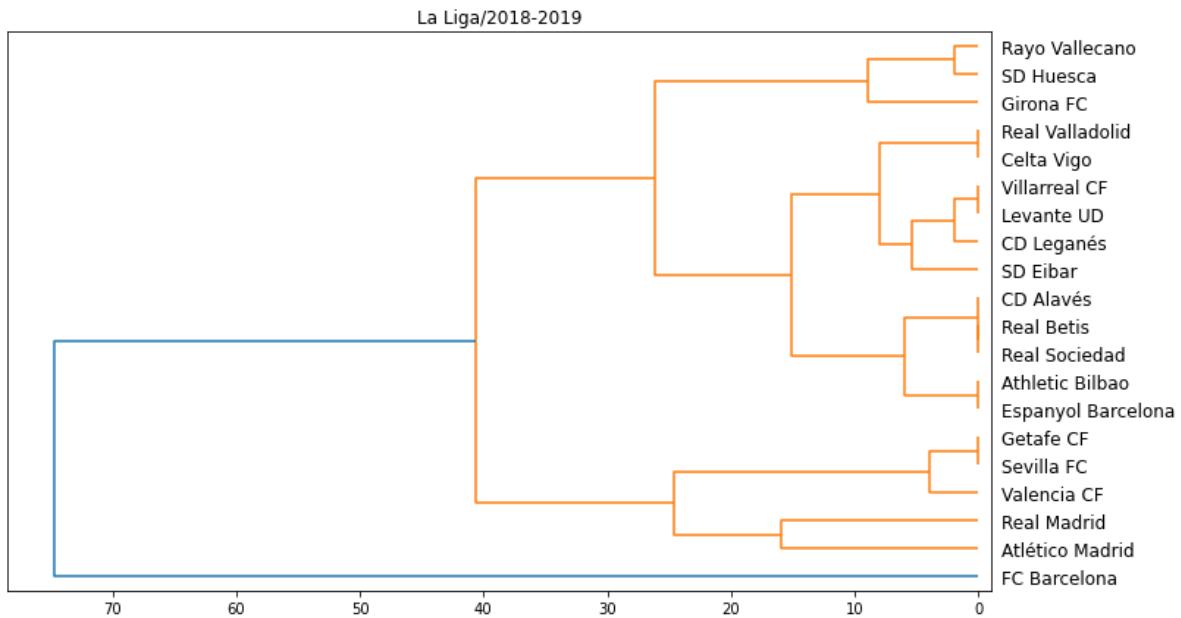


Figura 5.3: Dendrograma generado para “La Liga” al final de temporada ( $S_i(R)$ ) usando el agrupamiento enlace media o promedio [3] como método y la distancia de Manhattan como métrica [4].

Como puede verse, tomando la distancia de Manhattan los resultados varían en muy pocos centímetros con respecto al anterior dendrograma en el que se hacía uso de la distancia euclíadiana (4.2). Esto tiene sentido, pues la diferencia entre cómo se calcula una y otra tiene muy poco efecto en el resultado al utilizar distancias tan pequeñas.

Sin embargo, se observa que cambiando el método que el artículo en cuestión [1] dice utilizar por otro diferente, entonces se consigue un resultado prácticamente idéntico al que se debería obtener (el de la imagen 5.2). Esto ocurre independientemente de cuál de las 2 métricas se escoja para medir la distancia, pues como se acaba de ver, que se utilice la distancia euclíadiana o la de Manhattan es irrelevante en el resultado final. El dendrograma en cuestión con este nuevo método es el siguiente (imagen 5.4):

## CAPÍTULO 5. CONCLUSIONES

---

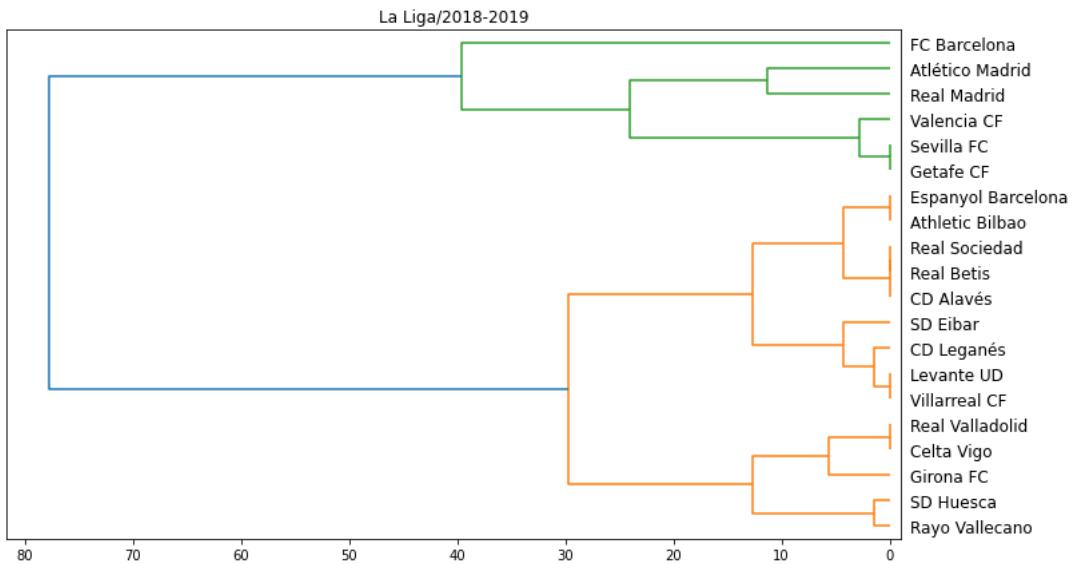


Figura 5.4: Dendrograma generado para “La Liga” al final de temporada ( $S_i(R)$ ) usando el agrupamiento de máximo o completo enlace [5] como método.

En este último dendrograma se ha utilizado el método del agrupamiento de máximo o completo enlace [5] en vez del agrupamiento enlace media o promedio [3] que el artículo [1] asegura utilizar. De hecho, si se secciona en 5 grupos igual que hace el artículo, se logran unos resultados idénticos a los obtenidos en la figura 5.2 y que se presentan a continuación en la imagen 5.5.

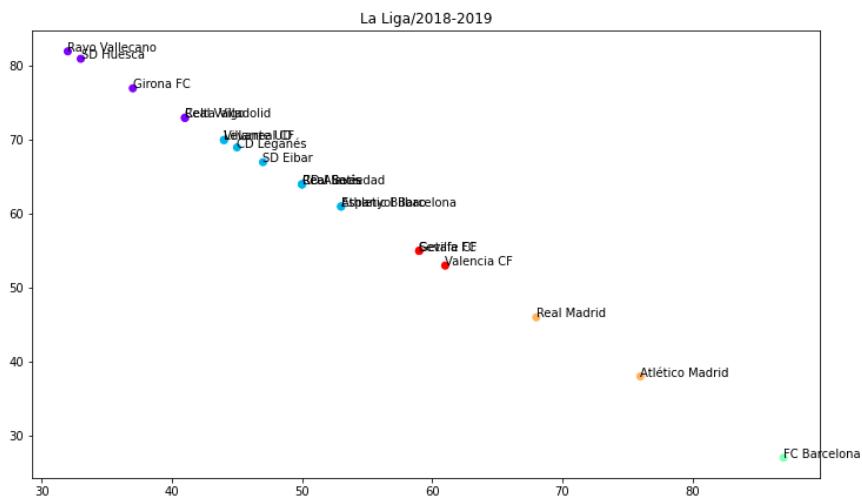


Figura 5.5: Clústeres generados a partir de seccionar el dendrograma de “La Liga” en 5 grupos. El dendrograma ha utilizado el agrupamiento de máximo o completo enlace [5] como método.

De hecho, esto es algo que ocurre en el resto de ligas también. Si se van probando los diferentes métodos (parámetro “method”) proporcionados por las funciones *linkage* [32] y *AgglomerativeClustering* [37], se observa que para conseguir los mismos resultados que en el artículo en cuestión [1] hay que usar las siguientes métricas [32] (5.1):

$$\begin{aligned} \textit{La Liga} (2018/2019) &\rightarrow \textit{method} = \text{‘complete’}; \\ \textit{Premiership} (2018/2019) &\rightarrow \textit{method} = \text{‘weighted’}; \\ \textit{Serie A} (2018/2019) &\rightarrow \textit{method} = \text{‘weighted’}; \\ \textit{Ligue 1} (2018/2019) &\rightarrow \textit{method} = \text{‘complete’}; \\ \textit{Série A} (2019) &\rightarrow \textit{method} = \text{‘average’}; \\ \textit{J1 League} (2019) &\rightarrow \textit{method} = \text{‘ward’}. \end{aligned} \tag{5.1}$$

Esto parece indicar que o bien la librería *scipy* [30] hace los cálculos de forma incorrecta (cosa poco probable al ser una librería de código abierto desarrollada por un amplio grupo de programadores [105]), o bien cabe la posibilidad de que los autores del artículo [1] se olvidaron de añadir este detalle. En cualquiera de los casos no se puede saber con certeza, por lo que el análisis de por qué ocurre esto se deja para futuros trabajos (5.1).

Continuando con el Escalamiento Multidimensional, el artículo [1] se obtienen los siguientes gráficos con el proceso de MDS (figura 5.6 tomada de la sección 3 de [1])

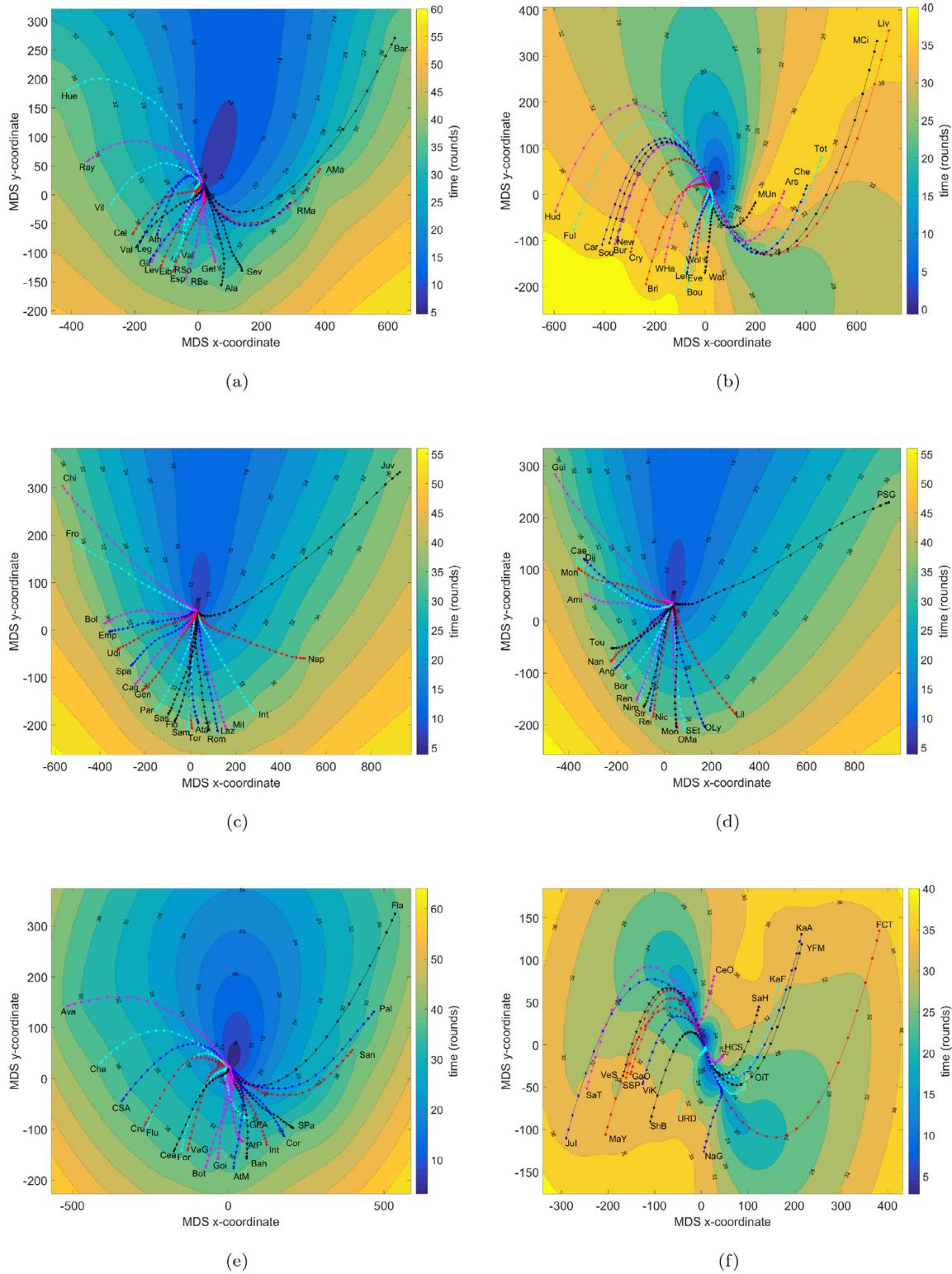


Figura 5.6: Los gráficos resultantes del MDS con dimensión  $M = 2$  de  $S_i(k) = P_i(k) + \imath Q_i(k)$ ,  $0 \leq k \leq k_r$ ,  $k_r = 0, \dots, R$ , con la distancia de Manhattan  $d_M$  para las ligas: (a) “La Liga”; (b) “Premiership”; (c) “Serie A”; (d) “Ligue 1”; (e) “Série A”; (f) “J1 League”. La tercera dimensión es calculada por RBI en función del tiempo (rondas).

Estos gráficos distan mucho de los obtenidos en este TFG (imagen 4.4).

Esta amplia diferencia se puede deber a un proceso completamente distinto para el escalamiento multidimensional, pues con unas desigualdades tan grandes no basta con solo cambiar el “random\_state”. No solo eso, sino que los gráficos del artículo cuentan con curvas de nivel [106] que aquí se ignoraron por claridad en los resultados.

Explicar y clarificar por qué hay una diferencia tan grande entre ambos resultados se deja a futuros trabajos también.

Tras representar los gráficos del poder complejo y del MDS, el artículo [1] se calcula la tabla con las medidas de competitividad (tabla 5.7 tomada la sección 4 de [1]), donde se ordenan las ligas de menor a mayor  $HICB$ , es decir, de mayor a menor competitividad:

|               | $b$   |           | $H$   |           | $\sigma$ | $HICB$  |
|---------------|-------|-----------|-------|-----------|----------|---------|
|               | $b_S$ | 2-dim MDS | $H_S$ | 2-dim MDS |          |         |
| ‘J1 League’   | 1.241 | 0.149     | 1.349 | 0.081     | 0.170    | 106.301 |
| ‘La Liga’     | 1.239 | 0.147     | 1.345 | 0.078     | 0.191    | 106.928 |
| ‘Premiership’ | 1.180 | 0.140     | 1.318 | 0.061     | 0.208    | 109.095 |
| ‘Ligue 1’     | 1.238 | 0.143     | 1.332 | 0.062     | 0.210    | 109.307 |
| ‘Série A’     | 1.180 | 0.142     | 1.316 | 0.059     | 0.217    | 109.740 |
| ‘Serie A’     | 1.179 | 0.140     | 1.315 | 0.058     | 0.220    | 111.652 |

Figura 5.7: Tabla con las medidas de competitividad  $b_S$ ,  $b_{MDS}$ ,  $H_S$ ,  $H_{MDS}$ ,  $\sigma$  y  $HICB$  (nombradas de izquierda a derecha) para la “J1 League”, “La Liga”, “Premiership”, “Ligue 1”, “Série A” y “Serie A”, ordenadas de menor a mayor  $HICB$  y  $\sigma$ .

En esta tabla se encuentran varias diferencias con la tabla de medidas obtenida en este TFG (tabla 4.5), principalmente en la dimensión fractal ( $b$ ) y la entropía ( $H$ ) que son distintas en cada liga. Esto seguramente se debe a las diferentes imágenes obtenidas y posibles diferencias en la implementación. Sin embargo, estas diferencias son esperadas, ya que como se acaba de ver, los gráficos del MDS son muy distintos y, además de eso, un pequeño cambio en la implementación de los métodos de obtención de la dimensión fractal y de la entropía haría cambiar considerablemente sus resultados. Las diferencias que son inesperadas son las que se dan en la desviación típica ( $\sigma$ ) y el índice de balance competitivo de Herfindahl-Hirschman ( $HICB$ ), pues estas se basan en los datos obtenidos y se tiene la certeza de que estos son los mismos, pues ambos provienen de la misma fuente: [www.worldfootball.net](http://www.worldfootball.net) [18].

## CAPÍTULO 5. CONCLUSIONES

---

Para la desviación típica ( $\sigma$ ) se observan valores distintos en la “Série A” (Brasil), donde para el artículo [1] el valor final es de 0,217 (5.7), sin embargo el aquí calculado es de 0,198 (4.5). Y en el caso del índice de balance competitivo de Herfindahl-Hirschman (*HICB*), todos los valores de las ligas son iguales menos para la “Premiership” (Inglaterra), donde el artículo [1] se obtiene 109,095 (5.7), mientras que en este TFG el resultado final es de 114,675 (4.5)

Una vez revisados y comprobado que son correctos todos los cálculos y los datos tomados por este TFG, solo me queda pensar que o bien hay algún fallo en la implementación hecha, o el fallo está en el artículo original [1]. Lo que me hace inclinarme a pensar que pueda estar del lado del artículo [1] es que, como se puede observar, a pesar de que para estas 2 medidas la mayoría de resultados coinciden, hay un par que no, pero en este TFG se aplica la misma metodología para todas las ligas, tanto para extraer datos como para sus cálculos. Como con la misma metodología se obtienen unos resultados iguales y otros diferentes, esto hace pensar que el error no está en la implementación aquí utilizada, pues de estar ahí habría diferencias en todas las ligas de forma sistemática, lo que significa que quizás los autores del artículo [1] hicieran estos cálculos a mano. Es por esto por lo que, de estar el error en el artículo [1], es posible que fuera causado por errores humanos y por no automatizar los cálculos, razón por la que el contenido de este TFG adquiere más valor aún.

Esto conlleva inevitablemente unas representaciones de las gráficas de las medidas de competitividad diferentes respecto a las ya calculadas en la imagen 4.6, evidentemente. Las gráficas de las medidas de competitividad del artículo serán tal que (figura 5.8 tomada de la sección 4 de [1]):

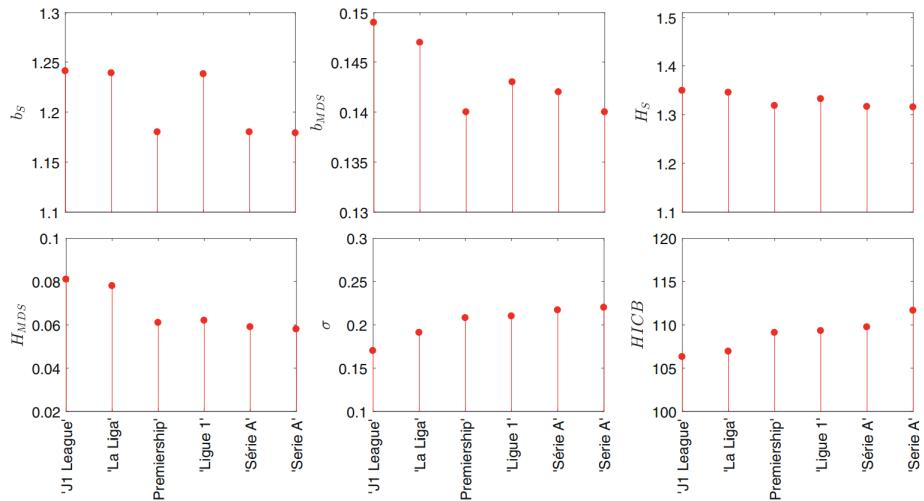


Figura 5.8: Los resultados obtenidos de  $b_S$ ,  $b_{MDS}$ ,  $H_S$ ,  $H_{MDS}$ ,  $\sigma$  y  $HICB$ . Las ligas están en orden descendente de competitividad según las medidas clásicas  $\sigma$  y  $HICB$ . Resultados basados en la tabla 5.7.

De todas maneras puede existir la posibilidad de que aun así se mantenga la correlación entre las variables, por lo que se comprobará esto atendiendo al gráfico representativo de la correlación de Pearson del artículo (figura 5.9 tomada de la sección 4 de [1]):

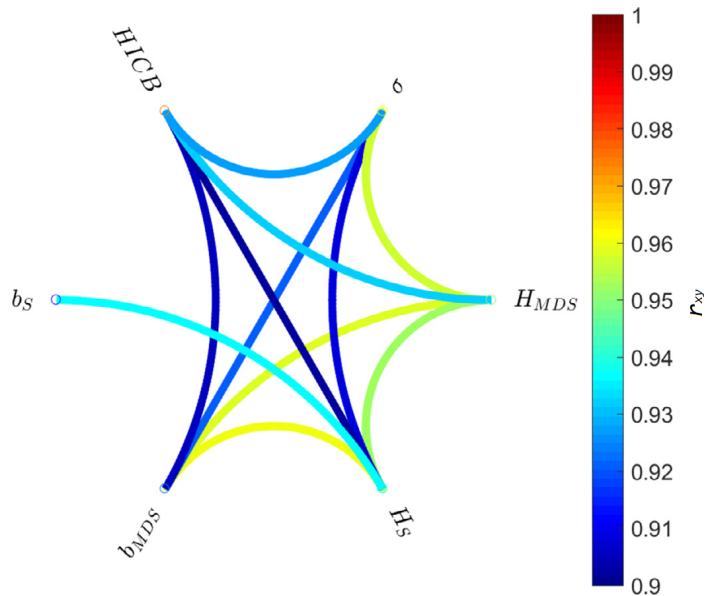


Figura 5.9: Coeficiente de correlación de Pearson  $r_{xy}$  entre las medidas de competitividad  $b_S$ ,  $b_{MDS}$ ,  $H_S$ ,  $H_{MDS}$ ,  $\sigma$  y  $HICB$ , para la “J1 League”, “La Liga”, “Premiership”, “Ligue 1”, “Série A” y “Serie A”. Los arcos denotan los valores donde  $|r_{xy}| \geq 0,9$  y los colores son proporcionales a  $r_{xy}$  según el mapa de colores proporcionado a la derecha.

En esta gráfica se usarán diferentes colores y solo mostrará los arcos donde  $|r_{xy}| \geq 0,9$ , es decir, los arcos donde la medida  $x$  tiene una correlación en valor absoluto mayor o igual a 0,9 con la medida  $y$ , es decir, una correlación muy alta. Para ponerlo en perspectiva, generalmente ya se considera una correlación alta si  $|r_{xy}| \geq 0,5$ , y baja si  $|r_{xy}| < 0,3$  [107].

Como se puede observar atendiendo a la imagen 4.8 vista en el capítulo 4 y comparándola con esta 5.8, los resultados obtenidos son completamente diferentes, coincidiendo únicamente en 3 arcos:  $b_{MDS} - H_{MDS}$ ,  $b_{MDS} - H_S$  y  $H_S - H_{MDS}$ . No solo esto, sino que la matriz de correlación obtenida en este TFG (4.7) varía con únicamente cambiar el valor del “random\_state” en la función `sklearn.manifold.MDS` [42], lo que indica que es normal que se den estas diferencias entre los resultados de uno y otro.

Con todo esto se puede concluir que a pesar de las diferencias en los resultados, se han informatizado con éxito todos los procesos del artículo: “Modeling and visualizing competitiveness in soccer leagues” [1].

## 5.1. Trabajos futuros

A pesar de haber explicado y expandido sobre todo el proceso de informatización del artículo: “Modeling and visualizing competitiveness in soccer leagues” [1] en todos sus puntos, hay, al menos, un total de 3 materias de las que se podrían realizar futuros trabajos para mejorar lo detallado en este TFG:

1. Diferencias en resultados.
2. Desarrollo de un aplicación.
3. Análisis en profundidad de los cálculos y resultados.

La primera (1) es la que se ha mencionado ya, el análisis de las diferencias en los resultados obtenidos frente a los del artículo. El problema principal de esto es que se tiene alguna intuición de por qué pueden estar ocurriendo estas disimilitudes en la mayoría de los casos, pero lo que se debería hacer para comprobar con certeza por qué ocurren es ponerse en contacto con los autores del artículo original [1]. Una vez hecho esto, si hacen uso de código informático sería de gran utilidad analizarlo. Además, con su ayuda, no solo se podrían corregir todas aquellas diferencias, sino entenderlas y hasta mejorarlas.

En segundo lugar está la creación de una aplicación que facilite el uso de todo este software (2). La idea podría ser tan compleja como se quiera, desde una app que se obtenga todos los gráficos deseados marcados en el ciclo de diseño (3.2.3), hasta algo más simple como poder introducir una liga y una temporada, y con ello obtenerse todos los datos de puntos por jornadas de esta. Para la segunda idea existe un acercamiento sencillo que son los “Google Forms” [108], con ellos se podrían llenar los campos de estas ligas en un formulario personalizado, enviar los datos al Google Sheets “RastreaLigas”, obtenerse los resultados y que se actualice automáticamente el fichero Excel llamado “Conexion\_DatosLigas” de la misma forma que ya se ha explicado en la sección 3.3.1. De todas formas esa solución quedaría eclipsada por una que use una base de datos como *MySQL* [109], donde primero se descargarían con el método de Google Sheets aquí explicado absolutamente todas las ligas que se deseen descargar, almacenarlas todas, y simplemente acceder a la que uno desee cuando quiera sin necesidad de descargar los datos cada vez como en la solución de “Google Forms”. En este TFG se hace una solución similar pero no tan a gran escala, pues solo se obtienen los datos de 7 ligas, por lo que se utiliza Excel como herramienta para la base de datos

Y, en último lugar, está el análisis en profundidad de los cálculos y resultados (3). Este contenido es más propio de un TFG del Grado de Matemáticas, de hecho, el Trabajo de Fin de Grado en el Grado en Matemáticas llamado “Modelización, visualización y análisis de la competitividad en diferentes ligas de fútbol profesional” [2] realizado por mí actúa como predecesor y complemento de este, donde se han explicado de forma mucho más detallada las ideas mostradas y expresadas en el artículo: “Modeling and visualizing competitiveness in soccer leagues” [1]. De esta forma, combinando ambas disciplinas, con esta pareja de Trabajos de Fin de Grado, culmina de forma armoniosa mi doble titulación en Ingeniería Informática y Matemáticas.

# Bibliografía

- [1] A. M. Lopes and J. A. Machado, “Modeling and visualizing competitiveness in soccer leagues,” *Applied Mathematical Modelling*, vol. 92, pp. 136–148, 2021.
- [2] “TFG\_Matematicas\_(PabloRoldanPeigneux).pdf - Google Drive.” [Online]. Available: <https://drive.google.com/file/d/1K7TE76JEN6dAx3PwYj-252HF-ZLtG4hr/view?usp=sharing>
- [3] “UPGMA - Wikipedia.” [Online]. Available: <https://en.wikipedia.org/wiki/UPGMA>
- [4] E. Deza, M. M. Deza, M. M. Deza, and E. Deza, “Encyclopedia of Distances,” in *Encyclopedia of Distances*. Springer Berlin Heidelberg, 2009, pp. 1–583. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-642-00234-2\\_1](https://link.springer.com/chapter/10.1007/978-3-642-00234-2_1)
- [5] “Hierarchical Clustering — solver.” [Online]. Available: <https://www.solver.com/xlminer/help/hierarchical-clustering-intro>
- [6] “Top 10 Most Popular Sports in The World.” [Online]. Available: <https://sportsshow.net/top-10-most-popular-sports-in-the-world/>
- [7] “fútbol — Definición — Diccionario de la lengua española — RAE - ASALE.” [Online]. Available: <https://dle.rae.es/f%C3%BAtbol>
- [8] *Pepe - Perfil del jugador — Transfermarkt.* [Online]. Available: <https://www.transfermarkt.es/raul/profil/spieler/7349>
- [9] “Raúl - Selección — Transfermarkt.” [Online]. Available: <https://www.transfermarkt.es/raul/nationalmannschaft/spieler/7349>

- [10] “Three points for a win - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Three\\_points\\_for\\_a\\_win#Year\\_of\\_adoption\\_of\\_three\\_points\\_for\\_a\\_win](https://en.wikipedia.org/wiki/Three_points_for_a_win#Year_of_adoption_of_three_points_for_a_win)
- [11] “La FIFA obliga a todas las ligas a conceder tres puntos por victoria — Deportes — EL PAÍS.” [Online]. Available: [https://elpais.com/diario/1994/10/29/deportes/783385203\\_850215.html](https://elpais.com/diario/1994/10/29/deportes/783385203_850215.html)
- [12] “WALKOVER — Definition of WALKOVER by Oxford Dictionary on Lexico.com also meaning of WALKOVER.” [Online]. Available: <https://www.lexico.com/definition/walkover>
- [13] The FA, “Law 3 The players,” 2019. [Online]. Available: <https://www.thefa.com/football-rules-governance/lawsandrules/laws/football-11-11/law-3---the-players> <http://www.thefa.com/football-rules-governance/lawsandrules/laws/football-11-11/law-3---the-players>
- [14] R. Criado, E. García, F. Pedroche, and M. Romance, “A new method for comparing rankings through complex networks: Model and analysis of competitiveness of major european soccer leagues,” *Chaos*, vol. 23, no. 4, 2013.
- [15] “Como calculan las casas de apuestas las cuotas.” [Online]. Available: <https://ekuatio.com/como-calculan-las-casas-de-apuestas-las-cuotas/>
- [16] E. E. País, “La casa de apuestas que no pudo comprar la fe de los más leales fans del Leicester,” may 2016. [Online]. Available: [https://verne.elpais.com/verne/2016/05/03/articulo/1462286064\\_102841.html](https://verne.elpais.com/verne/2016/05/03/articulo/1462286064_102841.html)
- [17] “Ingeniería Informática - Universidad Rey Juan Carlos.” [Online]. Available: <https://www.urjc.es/estudios/grado/628-ingeneria-informatica#competencias>
- [18] “worldfootball.net.” [Online]. Available: <https://www.worldfootball.net/>
- [19] “Google Sheets - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Google\\_Sheets](https://en.wikipedia.org/wiki/Google_Sheets)
- [20] “IMPORTXML - Ayuda de Editores de Documentos.” [Online]. Available: <https://support.google.com/docs/answer/3093342?hl=es>

- [21] “Saving data in Google Sheets with Google Apps Script -.” [Online]. Available: <https://www.benlcollins.com/spreadsheets/saving-data-in-google-sheets/>
- [22] “Custom Functions in Google Sheets — Apps Script — Google Developers.” [Online]. Available: <https://developers.google.com/apps-script/guides/sheets/functions>
- [23] “XPath - Wikipedia, la enciclopedia libre.” [Online]. Available: <https://es.wikipedia.org/wiki/XPath>
- [24] “Lenguaje de marcado - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Lenguaje\\_de\\_marcado](https://es.wikipedia.org/wiki/Lenguaje_de_marcado)
- [25] “XML - Wikipedia.” [Online]. Available: <https://en.wikipedia.org/wiki/XML>
- [26] “pandas.DataFrame — pandas 1.2.5 documentation.” [Online]. Available: <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html>
- [27] “NumPy - Wikipedia.” [Online]. Available: <https://en.wikipedia.org/wiki/NumPy>
- [28] “math — Mathematical functions — Python 3.9.5 documentation.” [Online]. Available: <https://docs.python.org/3/library/math.html>
- [29] “cmath — Mathematical functions for complex numbers — Python 3.9.5 documentation.” [Online]. Available: <https://docs.python.org/3/library/cmath.html>
- [30] “SciPy API — SciPy v1.7.0 Manual.” [Online]. Available: <https://docs.scipy.org/doc/scipy/reference/>
- [31] “scipy.cluster.hierarchy.dendrogram — SciPy v1.7.0 Manual.” [Online]. Available: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.dendrogram.html>
- [32] “scipy.cluster.hierarchy.linkage — SciPy v1.7.0 Manual.” [Online]. Available: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html>
- [33] “Cluster analysis - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Cluster\\_analysis](https://en.wikipedia.org/wiki/Cluster_analysis)
- [34] “scikit-learn - Wikipedia.” [Online]. Available: <https://en.wikipedia.org/wiki/Scikit-learn>

- [35] “skimage — skimage v0.19.0.dev0 docs.” [Online]. Available: <https://scikit-image.org/docs/dev/api/skimage.html>
- [36] “Pyplot tutorial — Matplotlib 3.4.2 documentation.” [Online]. Available: <https://matplotlib.org/stable/tutorials/introductory/pyplot.html>
- [37] “sklearn.cluster.AgglomerativeClustering — scikit-learn 0.24.2 documentation.” [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html>
- [38] “Matriz de correlación - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Matriz\\_de\\_correlaci%C3%B3n](https://es.wikipedia.org/wiki/Matriz_de_correlaci%C3%B3n)
- [39] “seaborn: statistical data visualization — seaborn 0.11.1 documentation.” [Online]. Available: <https://seaborn.pydata.org/>
- [40] M. Reddy, *API Design for C++*. Elsevier Science, 2011. [Online]. Available: <https://books.google.es/books?id=IY29LyIT85wC>
- [41] “World Football - Football API.” [Online]. Available: <https://football-api.com/plans/world-football/>
- [42] “sklearn.manifold.MDS — scikit-learn 0.24.2 documentation.” [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.manifold.MDS.html>
- [43] “Matriz transpuesta,” Tech. Rep.
- [44] “El ciclo de vida de un sistema de información,” Tech. Rep. [Online]. Available: <http://elvex.ugr.es/>
- [45] “RastreaLigas - Hojas de cálculo de Google.” [Online]. Available: <https://docs.google.com/spreadsheets/d/14dhiUC-nFACKjmKeVEHhFJyXjJEVgFuvJDwCIDPnaoQ/edit#gid=1882281593>
- [46] G. Weyenberg and R. Yoshida, “Reconstructing the Phylogeny: Computational Methods,” in *Algebraic and Discrete Mathematical Methods for Modern Biology*. Elsevier, mar 2015, pp. 293–319.

- [47] “Coeficiente de correlación de Pearson - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Coeficiente\\_de\\_correlaci%C3%B3n\\_de\\_Pearson](https://es.wikipedia.org/wiki/Coeficiente_de_correlaci%C3%B3n_de_Pearson)
- [48] “Tema 7. Escalamiento multidimensional.” [Online]. Available: [http://eio.usc.es/eipc1/BASE/BASEMASTER/FORMULARIOS-PHP/MATERIALESMASTER/Mat\\_14\\_master0809multi-tema7.pdf](http://eio.usc.es/eipc1/BASE/BASEMASTER/FORMULARIOS-PHP/MATERIALESMASTER/Mat_14_master0809multi-tema7.pdf)
- [49] “2.5 Variables discretas y continuas — Estadística Básica Edulcorada.” [Online]. Available: <https://bookdown.org/aquintela/EBE/variables-discretas-y-continuas.html>
- [50] J. M. Almendros-Jiménez and L. Iribarne, “Describing use-case relationships with sequence diagrams,” *Computer Journal*, vol. 50, no. 1, pp. 116–128, jan 2007.
- [51] “Caso de uso - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Caso\\_de\\_uso](https://es.wikipedia.org/wiki/Caso_de_uso)
- [52] G.ões@, “FATTO CONSULTORIA Y SISTEMAS Modelado y especificación de caso de uso,” Tech. Rep., 2019. [Online]. Available: <http://www1.fattocs.com/files/es/presentaciones/CasodeUso-04-2019-GuilhermeSimoes.pdf>
- [53] A. Cockburn, *Writing effective use cases*. Addison-Wesley, 2001.
- [54] “¿Caso de Uso?”
- [55] “Tipos de relaciones en diagramas de casos de uso. UML. — Blog SEAS.” [Online]. Available: <https://www.seas.es/blog/informatica/tipos-de-relaciones-en-diagramas-de-casos-de-uso-uml/>
- [56] U. d. V. Departamento de Informática and U. d. V. Departamento de Informática, *Casos de Uso*. [Online]. Available: <https://web.archive.org/web/20160705162936/http://www.infor.uva.es/~chernan/Ingenieria/Teoria/Tema3D.pdf>
- [57] “uml — ¿Cuál es la diferencia entre incluir y extender en el diagrama de casos de uso?” [Online]. Available: <https://www.it-swarm-es.com/es/uml/cual-es-la-diferencia-entre-incluir-y-extender-en-el-diagrama-de-casos-de-uso/968968207/>

- [58] “Diagrama de secuencia - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Diagrama\\_de\\_secuencia](https://es.wikipedia.org/wiki/Diagrama_de_secuencia)
- [59] “Diagrama de estado UML: estructura y funciones - IONOS.” [Online]. Available: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/diagrama-de-estado-uml/>
- [60] “Serie A 2018/2019 » 1. Round.” [Online]. Available: <https://www.worldfootball.net/schedule/ita-serie-a-2018-2019-spieltag/1/>
- [61] “pandas.read\_excel — pandas 1.3.0 documentation.” [Online]. Available: [https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read\\_excel.html](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_excel.html)
- [62] “excel - extract multiple tables from spreadsheet using python - Stack Overflow.” [Online]. Available: <https://stackoverflow.com/questions/43173297/extract-multiple-tables-from-spreadsheet-using-python>
- [63] “copy — Shallow and deep copy operations — Python 3.9.6 documentation.” [Online]. Available: <https://docs.python.org/3/library/copy.html>
- [64] “Python Shallow Copy and Deep Copy (With Examples).” [Online]. Available: <https://www.programiz.com/python-programming/shallow-deep-copy>
- [65] “Hierarchical clustering (scipy.cluster.hierarchy) — SciPy v1.7.0 Manual.” [Online]. Available: <https://docs.scipy.org/doc/scipy/reference/cluster.hierarchy.html>
- [66] “2.3. Clustering — scikit-learn 0.24.2 documentation.” [Online]. Available: <https://scikit-learn.org/stable/modules/clustering.html>
- [67] “Visualize multidimensional datasets with MDS — by Gianluca Malato — Towards Data Science.” [Online]. Available: <https://towardsdatascience.com/visualize-multidimensional-datasets-with-mds-64d7b4c16eaa>
- [68] “numpy.ndarray.flatten — NumPy v1.21 Manual.” [Online]. Available: <https://numpy.org/doc/stable/reference/generated/numpy.ndarray.flatten.html>
- [69] “numpy.ndarray.tolist — NumPy v1.21 Manual.” [Online]. Available: <https://numpy.org/doc/stable/reference/generated/numpy.ndarray.tolist.html>

- [70] “Fractal dimension computing · GitHub.” [Online]. Available: <https://gist.github.com/viveksck/1110dfca01e4ec2c608515f0d5a5b1d1>
- [71] “greycomatrix - skimage - Python documentation - Kite.” [Online]. Available: <https://www.kite.com/python/docs/skimage.feature.greycomatrix>
- [72] “statistics - What is the entropy of an image and how is it calculated? - Stack Overflow.” [Online]. Available: <https://stackoverflow.com/questions/50313114/what-is-the-entropy-of-an-image-and-how-is-it-calculated>
- [73] “Using a Gray-Level Co-Occurrence Matrix (GLCM) :: Analyzing and Enhancing Images (Image Processing Toolbox User’s Guide).” [Online]. Available: <http://matlab.izmiran.ru/help/toolbox/images/enhanc15.html>
- [74] “matplotlib.pyplot.imread — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.imread.html](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.imread.html)
- [75] “Module: color — skimage v0.19.0.dev0 docs.” [Online]. Available: <https://scikit-image.org/docs/dev/api/skimage.color.html>
- [76] T. Pawłowski, C. Breuer, and A. Hovemann, “Top Clubs’ Performance and the Competitive Situation in European Domestic Football Competitions,” *Journal of Sports Economics*, vol. 11, no. 2, pp. 186–202, 2010. [Online]. Available: <https://doi.org/10.1177/1527002510363100>
- [77] “numpy.corrcoef — NumPy v1.21 Manual.” [Online]. Available: <https://numpy.org/doc/stable/reference/generated/numpy.corrcoef.html>
- [78] J. M. A. Cabezas, J. M. Arias Cabezas, and I.áez@, *Aritmética y Álgebra*. Madrid: Grupo Editorial Bruño, Sociedad Limitada, 2008.
- [79] “Python abs().” [Online]. Available: <https://www.programiz.com/python-programming/methods/built-in/abs>
- [80] “matplotlib.pyplot.plot — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.plot.html#matplotlib.pyplot.plot](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html#matplotlib.pyplot.plot)

- [81] “matplotlib.pyplot.scatter — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.scatter.html#matplotlib.pyplot.scatter](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html#matplotlib.pyplot.scatter)
- [82] “matplotlib.pyplot.annotate — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.annotate.html#matplotlib.pyplot.annotate](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.annotate.html#matplotlib.pyplot.annotate)
- [83] “matplotlib.pyplot.legend — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.legend.html#matplotlib.pyplot.legend](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.legend.html#matplotlib.pyplot.legend)
- [84] “matplotlib.pyplot.xlabel — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.xlabel.html#matplotlib.pyplot.xlabel](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.xlabel.html#matplotlib.pyplot.xlabel)
- [85] “matplotlib.pyplot.ylabel — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.ylabel.html#matplotlib.pyplot.ylabel](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.ylabel.html#matplotlib.pyplot.ylabel)
- [86] “matplotlib.pyplot.xlim — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.xlim.html#matplotlib.pyplot.xlim](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.xlim.html#matplotlib.pyplot.xlim)
- [87] “matplotlib.pyplot.ylim — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.ylim.html#matplotlib.pyplot.ylim](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.ylim.html#matplotlib.pyplot.ylim)
- [88] “matplotlib.pyplot.figure — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.figure.html#matplotlib.pyplot.figure](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.figure.html#matplotlib.pyplot.figure)
- [89] “matplotlib.pyplot.show — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.show.html#matplotlib.pyplot.show](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.show.html#matplotlib.pyplot.show)
- [90] “matplotlib.pyplot.xticks — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.xticks.html#matplotlib.pyplot.xticks](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.xticks.html#matplotlib.pyplot.xticks)

- [91] “matplotlib.pyplot.axis — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.axis.html#matplotlib.pyplot.axis](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.axis.html#matplotlib.pyplot.axis)
- [92] “c - Calculate coordinates of a regular polygon’s vertices - Stack Overflow.” [Online]. Available: <https://stackoverflow.com/questions/3436453/calculate-coordinates-of-a-regular-polygons-vertices>
- [93] “matplotlib.patches.Patch — Matplotlib 3.4.2 documentation.” [Online]. Available: [https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.patches.Patch.html](https://matplotlib.org/stable/api/_as_gen/matplotlib.patches.Patch.html)
- [94] “seaborn.heatmap — seaborn 0.11.1 documentation.” [Online]. Available: <http://seaborn.pydata.org/generated/seaborn.heatmap.html#seaborn.heatmap>
- [95] “Bolton are not the first team to start a season with a points deduction — FourFourTwo.” [Online]. Available: <https://www.fourfourtwo.com/news/bolton-are-not-first-team-start-a-season-a-points-deduction>
- [96] “parseInt() - JavaScript — MDN.” [Online]. Available: [https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global\\_Objects/parseInt](https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/parseInt)
- [97] “Historia de la Liga española: Los primeros años (1929-1936) — rfef.es.” [Online]. Available: <https://www.rfef.es/noticias/rfef/historia-liga-espanola-1928-1936>
- [98] “Anexo:Estadísticas de la Primera División de España - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Anexo:Estad%C3%ADsticas\\_de\\_la\\_Primera\\_Divisi%C3%B3n\\_de\\_Espa%C3%A1a#Goles](https://es.wikipedia.org/wiki/Anexo:Estad%C3%ADsticas_de_la_Primera_Divisi%C3%B3n_de_Espa%C3%A1a#Goles)
- [99] “Python implementation to find any point symmetrical point about straight line - Programmer Sought.” [Online]. Available: <https://www.programmersought.com/article/47113667049/>
- [100] “python - How to find point closest to another in a group of points? - Stack Overflow.” [Online]. Available: <https://stackoverflow.com/questions/54929417/how-to-find-point-closest-to-another-in-a-group-of-points/54929852>

- [101] “math - How to calculate the coordinates of the line between two points in python? - Stack Overflow.” [Online]. Available: <https://stackoverflow.com/questions/43594646/how-to-calculate-the-coordinates-of-the-line-between-two-points-in-python>
- [102] “Bundesliga 2018/2019 » 34. Round.” [Online]. Available: <https://www.worldfootball.net/schedule/bundesliga-2018-2019-spieltag/>
- [103] “python - Find point-B on line by distance from point-A - Stack Overflow.” [Online]. Available: <https://stackoverflow.com/questions/18843469/find-point-b-on-line-by-distance-from-point-a>
- [104] “Problemas de proporcionalidad: proporcionalidad directa e inversa.” [Online]. Available: <https://www.smartick.es/blog/matematicas/recursos-didacticos/problemas-de-proporcionalidad/>
- [105] “SciPy Core Developer Guide — SciPy v1.7.0 Manual.” [Online]. Available: <https://docs.scipy.org/doc/scipy/dev/core-dev/index.html>
- [106] “Contour line - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Contour\\_line#cite\\_ref-1](https://en.wikipedia.org/wiki/Contour_line#cite_ref-1)
- [107] “Pearson’s Correlation Coefficient - Statistics Solutions.” [Online]. Available: <https://www.statisticssolutions.com/pearsons-correlation-coefficient/>
- [108] “Formularios de Google.” [Online]. Available: <https://docs.google.com/forms/u/0/>
- [109] “MySQL.” [Online]. Available: <https://www.mysql.com/>