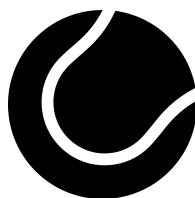




universidad  
de león



## **BreakPoint**

Sistema recomendador de raquetas  
y cordajes de tenis

Diego Simón González



# Índice

<b>MOTIVACIÓN</b>	<b>4</b>
<b>RESUMEN</b>	<b>4</b>
<b>OBJETIVOS</b>	<b>5</b>
<b>HERRAMIENTAS</b>	<b>5</b>
Neo4j	5
Utilidad	5
Certificados	5
Certificado "Overview"	6
Certificado "Querying With Cypher"	6
Certificado "Creating Nodes and Relationships"	6
<b>GRAFO DE CONOCIMIENTO</b>	<b>9</b>
Obtención de conjuntos de datos	9
Raquetas	9
Limpieza y formato de los datos	9
Importación de los datos	9
Esquema de los datos en el grafo	9
<b>ALGORITMOS</b>	<b>9</b>
Algoritmo colaborativo	10
<b>ANÁLISIS DE RESULTADOS</b>	<b>11</b>
Algoritmo colaborativo	11
Algoritmo de contenido	11
<b>ANÁLISIS CRÍTICO DEL SISTEMA</b>	<b>11</b>
Beneficios	11



Limitaciones	11
<b>EVOLUCIÓN</b>	<b>11</b>
<b>CONCLUSIONES Y APRENDIZAJE</b>	<b>11</b>
<b>BIBLIOGRAFÍA Y OTRAS FUENTES</b>	<b>12</b>

## 1. MOTIVACIÓN

En el mundo del tenis, la cantidad de raquetas y cordajes que se pueden adquirir en el mercado puede llegar a abrumar a los jugadores amateur que practican este deporte. Es por ello que, generalmente, requieren del consejo de algún experto u otra persona de un nivel superior que pueda darles las claves a la hora de encontrar un producto adecuado a sus necesidades.

De esta situación nace BreakPoint, un sistema recomendador de raquetas y cordajes que se apoya en un grafo de conocimiento creado con la tecnología Neo4j. Con esta aplicación web, cualquier usuario puede registrarse en la página para aportar sus valoraciones y así obtener una lista de raquetas y cordajes que pueden adaptarse a su juego.

Al ser una aplicación dirigida a un público que no tiene por qué tener conocimientos acerca del funcionamiento de una base de datos de este tipo u otras características del sistema, la interfaz gráfica debe ser sencilla de utilizar y atractiva a la vista.

## 2. RESUMEN

El objetivo principal de este proyecto es conseguir crear una aplicación web disponible para jugadores de tenis amateur que sirva como un recomendador de raquetas, teniendo en cuenta tanto sus gustos como sus características físicas. Esta aplicación consta de: una parte cliente creada con la tecnología React.js y una parte servidora a la que se conecta el frontend a través de peticiones HTTP, además de una base de datos en forma de grafo de conocimiento que corre en un servidor Cloud de Aura (específico para Neo4j). La parte servidora está escrita en Javascript y es la encargada de ejecutar los algoritmos de recomendación. Estos algoritmos serán:

### 1) Colaborativo

Basado en las valoraciones que ha hecho el usuario que está usando la aplicación en comparación con las valoraciones que han hecho otros usuarios previamente

### 2) Contenido

Dado un vector de características, se le asigna un grado de importancia a cada una de ellas y se ejecuta el algoritmo teniendo en cuenta las valoraciones que haya hecho el usuario actual, únicamente.

### 3) Híbrido

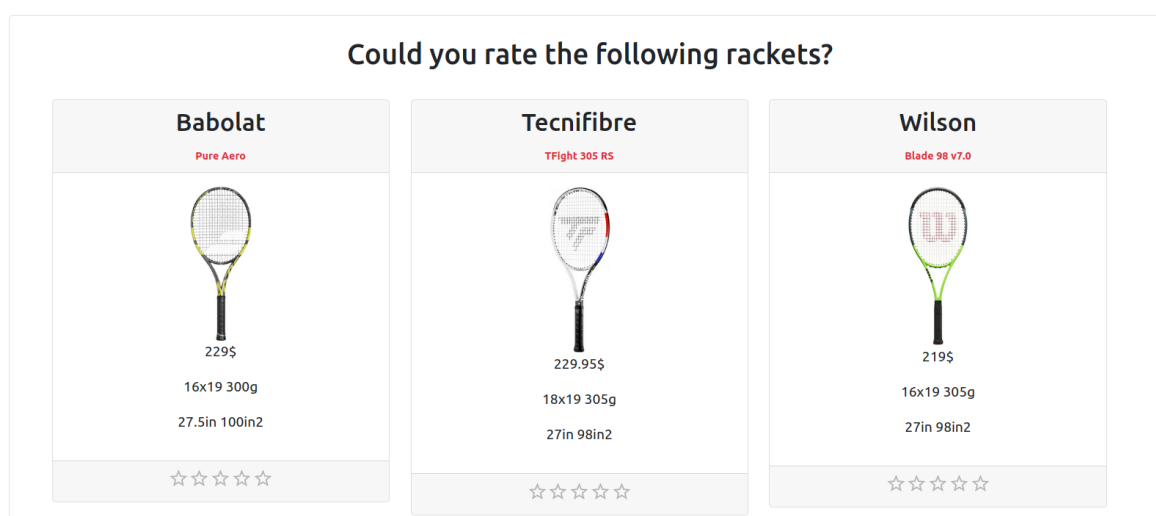
Una vez obtenidos los resultados de los algoritmos anteriores, estos se combinan, dando pesos específicos para obtener un conjunto final compuesto de 3 raquetas que se estiman adecuadas al juego del usuario.

### 4) Contenido con bonus para cordajes

Una vez obtenida una recomendación de raquetas mediante uno de los 3 algoritmos anteriores, el usuario puede rellenar una serie de campos para ejecutar un nuevo algoritmo basado en el de contenido, que tiene en cuenta nuevos parámetros para recomendar al usuario unos cordajes adecuados a su juego, sus gustos y a la raqueta escogida de entre las propuestas en el paso anterior.

Estos algoritmos se basan en una serie de conjuntos de datos. Uno de ellos incluye las características de las raquetas más populares del mercado; otro muestra información similar, pero referida a los cordajes; y el último almacena información referente a los jugadores más conocidos del mundo del tenis, incluyendo datos como la raqueta que usan o algunas de sus características físicas. Por último, hay uno que almacena información de usuarios que van a estar presentes en el sistema (creados artificialmente).

Con estos dos "datasets" se construirá un grafo en Neo4j, agregando usuarios, que tengan relaciones con los anteriores productos, para poder basar las decisiones del recomendador en este tipo de relaciones. En la **Fig. 1** se puede ver la pantalla principal de la aplicación, que se le presenta al usuario cuando accede al sitio web:



**Fig. 1:** Pantalla principal de BreakPoint. Fuente propia

### 3. OBJETIVOS

Un proyecto de estas características, realizado por una sola persona y en pocos meses, no puede aspirar a convertirse en un sistema revolucionario o implantable rápidamente, dado que se requiere la colaboración de: usuarios que puedan crear una red de valoraciones, trabajadores que mantengan al día la base de datos de raquetas a medida que estas salen al mercado, etc.

Sin embargo, la implementación propuesta puede sentar las bases para un proyecto a mayor escala que pueda interesar no sólo a jugadores amateur de tenis, sino a aquellas tiendas de deportes que venden raquetas y cordajes. Este sistema podría serles útil de cara a buscar las raquetas que mejor se adaptan a sus clientes. Además, un seguimiento de las sensaciones y valoraciones de estos conseguiría dar más precisión al recomendador y mejorar así el sistema globalmente.

A mayores, al tratarse de un proyecto académico, uno de los objetivos principales es conseguir que el alumno adquiriera un manejo fluido de las herramientas de las que dispone para elaborar un sistema inteligente. Principalmente, se adquiere facilidad para el uso de Neo4j y, en general, de grafos de conocimiento. Esta herramienta es ideal para un recomendador de estas propiedades, ya que los datos no se almacenan como en una base de datos al uso (tipo SQL), donde las relaciones se crean a la hora de extraer o introducir datos, sino que estas forman parte de la propia base de datos, formando así un grafo que da más información al usuario.

Además, este proyecto puede servir como una introducción a los algoritmos de inteligencia artificial que pueden ser utilizados para otro tipo de sistemas. No solo es necesaria investigación para conocer cuáles son los más adecuados, sino que también es necesaria creatividad para poder adaptarlos de la manera más eficiente y eficaz posible.

### 4. HERRAMIENTAS

#### 4.1. Neo4j

##### 4.1.1. ¿Por qué usar Neo4j?

Antes de responder a esta pregunta, debemos responder a otra más general: ¿por qué usar grafos de conocimiento?. La respuesta es sencilla. Las bases de datos convencionales no son capaces de almacenar relaciones, sino que las relaciones se crean a la hora de extraer, modificar o insertar datos, pero estas no persisten. Un grafo de conocimiento sí que almacena estas relaciones.

Esto es muy interesante para sistemas de recomendación, dado que son capaces de interpretar qué otras relaciones es posible o probable que se puedan crear, mediante algoritmos que deben ser implementados utilizando los datos de los que se dispone en el grafo.

Dentro de las diferentes tecnologías existentes para la gestión de bases de datos de grafos, Neo4j brinda unos niveles de escalabilidad y rendimiento que muchas otras no pueden alcanzar. Además, la facilidad de aprendizaje y la amplia comunidad de usuarios hacen de esta opción una de las más sencillas de implementar. Ver que Neo4j ya está siendo utilizado por gigantes empresas dentro de sectores muy distintos, véase IBM o eBay, nos indica que es una gran elección.

#### 4.1.2. Certificados

Entre los servicios que ofrece Neo4j, se encuentra la Graph Academy, que tiene como propósito servir como un método de introducción a los grafos de conocimiento y, en concreto, a Neo4j. En la Academy se puede participar en una serie de cursos. Cada uno de ellos, permite la descarga de un certificado que se le otorga a todo aquel que complete el curso en cuestión y conteste correctamente a todas las preguntas relativas al curso que se le muestran al final. Los certificados obtenidos se detallan a continuación:

##### 4.1.2.1. Certificado "Overview"

Primer curso dentro de la Graph Academy. Es un curso eminentemente teórico e introductorio. Se comentan las capacidades de la plataforma y se introducen algunos comandos útiles, entre los cuales yo destacaría: `$help commands`, `$help keys`, `$help MATCH`, `$sysinfo`, `$history`. Además, hay librerías que extienden las funciones de Cypher, como APOC

##### 4.1.2.2. Certificado "Querying With Cypher"

Segundo curso dentro de la Graph Academy. Combina aspectos teóricos y prácticos con pequeños cuestionarios para comprobar que se han adquirido los conocimientos necesarios para obtener el certificado. Se incluyen ejercicios que se pueden ejecutar en Sandbox, Aura o en Desktop para conseguir que el usuario resuelva diferentes problemas mediante sentencias en lenguaje Cypher, resaltando además las similitudes y diferencias con SQL. Tras realizar este curso, creo que Cypher podría considerarse, incluso, una versión mejorada de SQL, porque mantiene muchas de las funcionalidades, como las que provienen del uso de ORDER BY,

DISTINCT, WHERE..., añadiendo las ventajas de incluir en la base de datos las relaciones que hay entre los diferentes elementos.

#### 4.1.2.3. Certificado "Creating Nodes and Relationships"

Tercer curso de la Graph Academy de Neo4j. Se muestra cómo se crean nodos, relaciones y propiedades asociadas a ellos. Además de esto, se enseña a eliminar todos estos elementos, diferenciando, por ejemplo, en el caso de los nodos, aquellas sentencias que deben eliminar también las relaciones asociadas a ellos (DETACH DELETE). A mayores, se muestra la utilidad que presenta la sentencia MERGE, que permite crear nodos y relaciones siempre y cuando estos no existan previamente. En el caso de que se intentase crear elementos ya existentes en el grafo, solo se añadirían aquellos que no formen parte de él. De hecho, otra de las utilidades de MERGE es la capacidad de diferenciar dos flujos de ejecución de sentencias en función de si los elementos que se incluyen en el MERGE ya pertenecen al conjunto de datos del grafo (ON MATCH) o, por el contrario, se crean posteriormente (ON CREATE).

#### 4.1.2.4. Certificado "Constraints, Indexes and Best Practices"

Cuarto curso de la Graph Academy de Neo4j. Cuarto curso de la Graph Academy. Se muestra cómo se crean y eliminan constraints. Estas pueden ser de diferentes tipos:

- \*Uniqueness --> La misma propiedad en distintos nodos del mismo tipo no puede tener el mismo valor
- \*Existence --> Es obligatorio dar un valor a las propiedades indicadas
- \*Node key --> Define existence y uniqueness para múltiples propiedades de un nodo de un cierto tipo

Por otra parte, los índices sirven para dar mayor eficiencia a las operaciones realizadas con elementos que no tienen porqué tener valores distintos unos de otros. Hay tres tipos de índices: single-property, composite y full-text schema. Además, también se estudia el uso de parámetros que permitan ejecutar queries desde el exterior. Por último, se comentan algunas buenas prácticas, como: el uso de índices, la agregación previa a la cláusula RETURN, el empleo de DISTINCT y LIMIT...

## 4.2. Aura

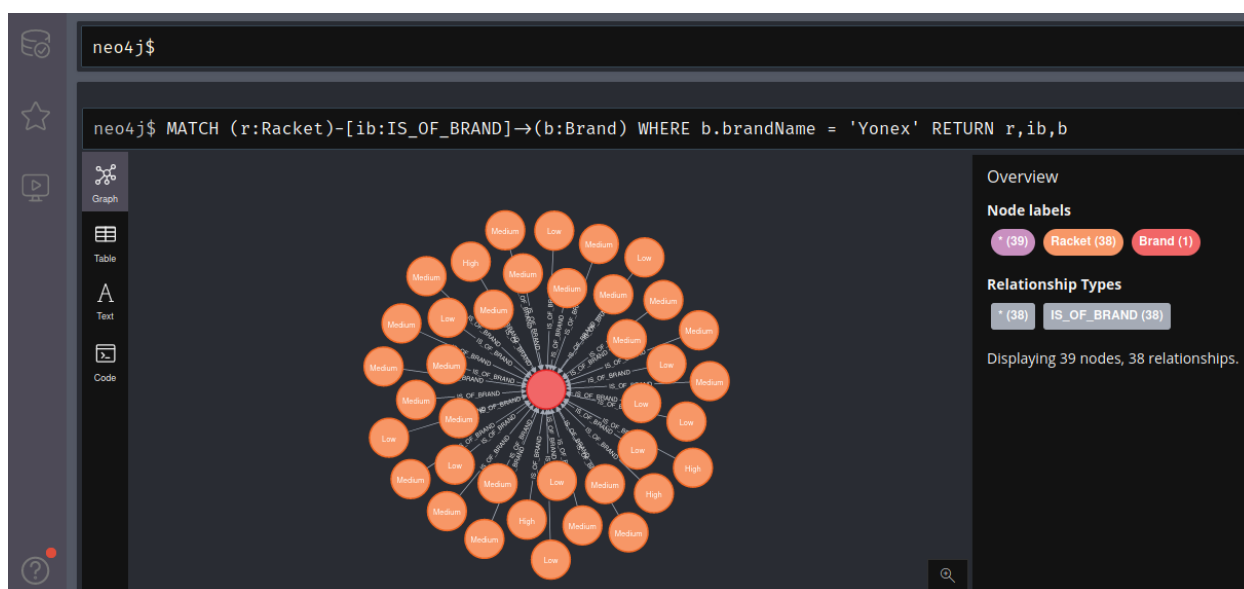
Aura es un servicio cloud ofrecido por el grupo Neo4j que permite la creación de bases de datos de grafos en servidores remotos, de manera que no es necesario ejecutar el sistema de Neo4j en local para cada inicio de la



aplicación web en desarrollo. Esto agiliza las tareas, además de que evita al desarrollador todo el consumo de recursos relacionado con los procesos de base de datos.

Otra ventaja de emplear Aura, es que en el apartado "Connect" se muestran ejemplos de código (en varios lenguajes), cuya finalidad es usar las librerías de Neo4j para realizar la conexión con la base de datos.

Pese a ser un servicio de pago, hay una opción gratuita que permite crear un pequeño proyecto (suficiente para la asignatura) de manera gratuita. Esta opción incluye un servicio web denominado Neo4j Browser a través del cual se puede manejar el grafo de conocimiento mediante las sentencias No SQL aprendidas en los cursos de la plataforma. En la **Fig. 2** se muestra este servicio web:



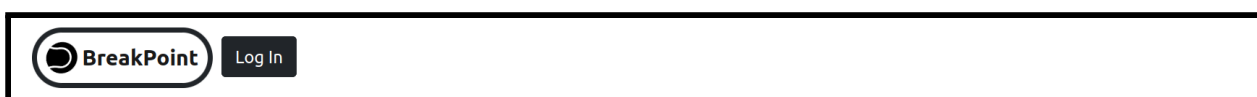
**Fig. 2:** Neo4j Browser for Aura. Fuente

### 4.3. Visual Studio Code

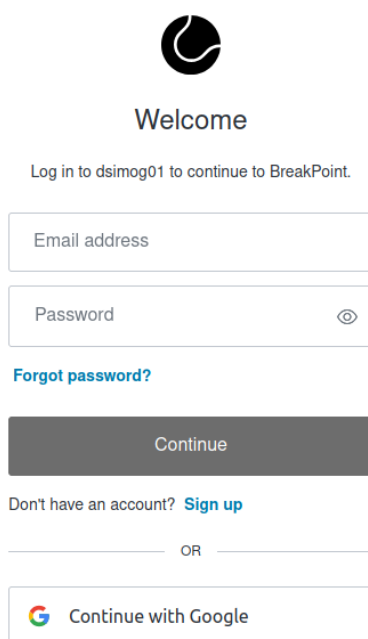
Este es el editor de texto gratuito empleado para desarrollar todo el código de la aplicación. Contiene tanto extensiones de cualquier tipo, como ayudas para todos los formatos de archivo que forman parte del proyecto, incluyendo .csv, .js, .cs... Además, he empleado GitHub Copilot como un ayudante a la hora de escribir código, pues es una herramienta basada en GPT-3, que es capaz de predecir con bastante exactitud el código siguiente a las líneas ya escritas en el momento de la predicción, usando para ello un motor creado con inteligencia artificial.

#### 4.4. Auth0

Auth0 es un servicio online que facilita la implementación de un login para una aplicación web creada con React.js. Mediante la creación de un usuario en su página web, se puede externalizar este servicio, permitiendo así que un usuario inicie sesión usando su correo electrónico o su cuenta de Google, de manera que no tendrá que registrarse específicamente para la nueva web. Un dato curioso, es el hecho de que la autenticación en Aura se realiza mediante este servicio. En la **Fig.3** se muestra la barra de navegación de la aplicación con el botón que permite acceder al "login". y en la **Fig. 4** se puede observar la pantalla de inicio de sesión generada usando esta herramienta:



**Fig. 3:** Barra de navegación de BreakPoint. Fuente propia



**Fig. 4:** Interfaz de inicio de sesión con Auth0. Fuente [1]

#### 4.5. React.js

React es una biblioteca para JavaScript que permite crear interfaces de usuario en forma de SPAs (Single Page Application). React

permite utilizar bootstrap y otras bibliotecas como complemento para facilitar el diseño y la implementación de los diferentes componentes que forman la aplicación final. No solo se pueden crear componentes estáticos y dinámicos, sino que también se pueden crear funciones que doten al sistema de lógica. Se ha decidido desarrollar la aplicación usando este framework porque es uno de los más utilizados en la actualidad, de modo que tiene una comunidad muy amplia y activa que facilita la creación de código. Además, existen otras bibliotecas que se complementan perfectamente con React para evitar la generación de componentes desde cero. Las animaciones, los contenedores, las imágenes, etc. se pueden disponer en la página de una manera sencilla y vistosa.

#### 4.6. Nginx

Nginx es un servidor web ligero que se puede instalar en diversos sistemas operativos de una manera muy sencilla para acceder a los ficheros presentes en el directorio raíz del servidor por medio de solicitudes HTTP. En el caso de este sistema, el servidor nginx es necesario para poder realizar la importación de datos al grafo de conocimiento, dado que, al estar este almacenado en el servicio cloud Aura, no se pueden subir los ficheros .csv directamente, sino que tiene que accederse a ellos mediante una petición HTTP. Para ello, se incluyen los diferentes ficheros en el servidor y se coloca la URL correspondiente a cada uno de ellos en el fichero .cypher correspondiente tras la orden "LOAD CSV".

#### 4.7. JStat

JStat es la biblioteca de JavaScript para cálculo de estadísticas por excelencia. En el caso de BreakPoint, se utiliza para obtener el coeficiente de correlación de Pearson, en el segundo paso del algoritmo colaborativo.

### 5. GRAFO DE CONOCIMIENTO

#### 5.1. Obtención de conjuntos de datos

Los conjuntos de datos que se han empleado para poblar el grafo de conocimiento, provienen de fuentes diversas, pero han sido tratados para formar una

base de datos que cumple las normas de cohesión y coherencia. El tratamiento que han sufrido los datos se detalla en el apartado 5.1.4.

### 5.1.1. Raquetas

Este es el conjunto de datos básico para el sistema. Proviene de un fichero en formato CSV descargado del sitio web de Kaggle (Fuente [2]). De las 15 columnas que componen este fichero, solo se mantendrán 11, que son las que se consideran relevantes para este caso de estudio en concreto. Las características que serán de utilidad, para cada raqueta, se enumeran a continuación: Marca, Precio, Tamaño de la cabeza, Longitud, Peso (sin cordaje), Tensión mínima del cordaje, Tensión máxima del cordaje, Ancho del marco, Flexibilidad, Potencia (salida de bola), Patrón de cuerdas. En la **Fig. 5** se observa una muestra del fichero, tras haber sido tratado:

modelID	brand	price_Dol	headSize_in2	length_in	weightUnstrung_g	minTension_kg	maxTension_kg	beamWidth_mm	flex	powerLevel	stringPattern_VxH
1	Head	239.95	98	27	315	21.77	25.85	20	64	Low	16x19
2	Tecnifibre	209.95	100	27	280	22.22	24.94	23	70	Medium	16x19
3	Tecnifibre	229.95	98	27	305	22.22	24.94	22.5	68	Low	18x19

**Fig. 5:** Conjunto de datos de raquetas tras tratamiento de datos. Fuente [2]

### 5.1.2. Cordajes

En este segundo "dataset" se almacena información sobre los cordajes más vendidos del mercado, confeccionado por mí mismo, tras un largo proceso de investigación. Además, se agregan los cordajes que forman parte del fichero CSV que se emplea en el siguiente apartado. En la **Fig. 6** se muestra el fichero tras el tratamiento:

modelID	brand	modelName	type	composition	gauge_mm
1	Babolat	RPM Blast 17	Monofilament	Co-polyester	1.25
2	Wilson	Luxilon Big Banger ALU Power 16	Monofilament	Aluminium	1.25
3	Tecnifibre	NRG2 17	Multifilament	Elastyl Fiber	1.24

**Fig. 6:** Conjunto de datos de cordajes tras tratamiento de datos. Fuente propia

### 5.1.3. Tenistas

El tercer conjunto de datos está formado por información acerca de los jugadores situados (actualmente o hace años) en lo más alto del ránking mundial. Para cada uno de ellos, se incluyen características físicas (estatura o

peso), además de datos acerca de la forma de golpear a la pelota que tiene cada uno de ellos. Aunque no se especifica en qué temporada se tomaron cada uno de los datos, se sabe que para jugador todos fueron tomados a la vez, de modo que la información es útil y relevante para el sistema. Para el grafo de conocimiento, esto aporta información relevante, puesto que la recomendación de raquetas y cordajes podrá no estar basada únicamente en gustos o valoraciones, sino también en las habilidades y características físicas del jugador que utilice la herramienta, usando como base la información conocida sobre jugadores del más alto nivel. En la fuente [3] se puede encontrar el dataset descrito. Al dataset original se le han agregado la estatura y el peso de cada uno de los jugadores, además del ID de la raqueta y el cordaje que usa cada uno de ellos. En la **Fig. 7** se visualiza dicho fichero:

playerID	player	height_m	weight_kg	racketLength_in	racketWeight_g	swingWeight_kgxc2	recoilWeight_kgxc2	polarization	mgr_i	stringID	backhandHands	racketID
1	Alexei Popyrin	1.96	78	27	338	370	194.3	0.57	19.48	9	2	87
2	Andre Agassi	1.8	80	27	366.6	351	162	0.44	21.22	10	2	1
3	Andy Murray	1.91	82	27	353	358	168	0.48	20.63	11	2	1

**Fig. 7:** Conjunto de datos de jugadores tras tratamiento de datos. Fuente [3]

#### 5.1.4. Usuarios

El cuarto y último conjunto de datos ha sido generado mediante un script en Python (userCreator.py). En el archivo CSV creado, se almacenan una serie de usuarios, cuyo nombre de usuario ha sido creado aleatoriamente mediante la biblioteca "random username". Además, de este dato, a cada usuario se le asigna una altura y un peso factibles según el IMC (Índice de Masa Corporal). Una vez almacenados los datos de los usuarios, se rellena el fichero con valoraciones. Los modelos de raquetas se representan en las columnas y los usuarios en las filas, de modo que el valor  $[i, j+2]$  representa a la puntuación que ha otorgado el usuario  $i$  a la raqueta  $j$ .

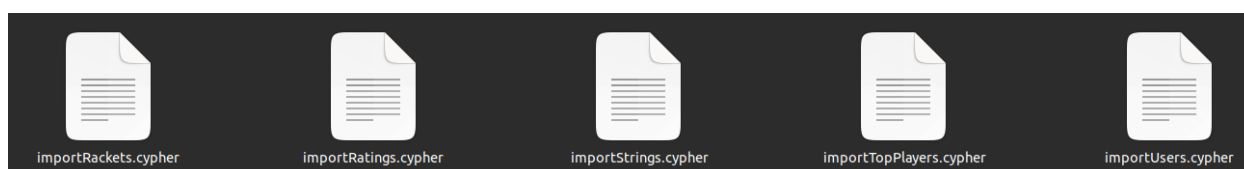
*NOTA: Se suma 3 a las columnas porque las dos primeras corresponden a la altura y el peso del usuario  $i$*

username	height	weight	racket1	racket2	racket3	racket4	racket5	racket6	racket7	racket8
superiorAntelope6	1.55	53	3	4	3	3	4	5	5	4
debonairOil9	1.76	69	4	5	4	4	5	4	4	5
solemnWildfowl6	1.67	62	4	5	4	4	5	4	4	5

**Fig. 8:** Conjunto de datos de usuarios tras tratamiento de datos. Fuente propia

## 5.2. Importación de los datos

Para que los datos presentes en los distintos ficheros CSV comentados en los apartados anteriores formen parte del grafo de conocimiento, es necesario desarrollar un script en formato "cypher", que en este caso está dividido en varios ficheros para diferenciar la funcionalidad de cada uno de ellos. Al estar usándose Aura, los ficheros se han subido a un servidor nginx corriendo en una Raspberry al que se accede mediante peticiones HTTP para descargarlos y usar los datos contenidos en ellos para formar el grafo de conocimiento en la nube. En la **Fig. 9** se muestran los ficheros en los que se divide el proceso de importación y en la **Fig. 10**, un fragmento de uno de ellos, que sirve como ejemplo de un archivo en formato .cypher:



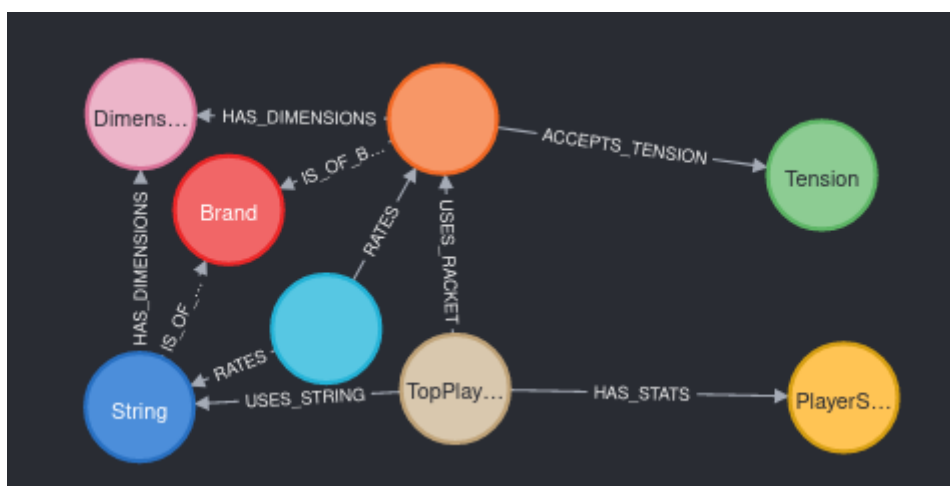
**Fig. 9:** Ficheros para importación de los conjuntos de datos al grafo. Fuente propia

```
LOAD CSV WITH HEADERS FROM 'http://81.43.174.98/BreakPoint/data/formattedFiles/formattedTennisRackets.csv' AS row
WITH row
WHERE row.modelID IS NOT NULL
MERGE (r:Racket {
    modelID: toInteger(row.modelID),
    price_Dol: toFloat(row.price_Dol),
    flex: toInteger(row.flex),
    powerLevel: row.powerLevel,
    stringPattern_VxH: row.stringPattern_VxH
})
```

**Fig. 10:** Fragmento del fichero 'importRackets.cypher'. Fuente propia

## 5.3. Esquema de los datos en el grafo

Tras importar todos los conjuntos de datos anteriores, creando nodos, propiedades y relaciones, el grafo sigue el esquema definido en la **Fig.11**. Los nodos principales son: el naranja (raquetas), el azul oscuro (cordajes), el azul claro (usuarios) y el marrón (jugadores profesionales). El resto de nodos sirve, principalmente, como información adicional a los anteriores y como elemento cohesivo entre nodos principales. Los usuarios tienen la capacidad de valorar tanto raquetas como cordajes. Esto se refleja en el grafo con relaciones 'RATES' y los jugadores profesionales usan las raquetas y los cordajes que se indican mediante las relaciones 'USES\_RACKET' y 'USES\_STRING'.



**Fig. 11:** Esquema del grafo de conocimiento. Fuente propia

## 6. ALGORITMOS

Los algoritmos básicos que permiten el funcionamiento del sistema son el colaborativo y el de contenido. Estas han sido las elecciones debido a que, al ser muy populares entre los sistemas recomendadores, están mejor documentados y hay una comunidad mayor que se dedica a su implementación. Por tanto, son más sencillos y efectivos. Además, las operaciones que se han de realizar para desarrollarlos, son bastante básicas, lo que permite que estos se puedan ejecutar en sistemas que no requieren un gran potencial de procesamiento.

### 6.1. Algoritmo colaborativo

#### 6.1.1. Creación de una tabla de valoraciones

En la tabla de valoraciones se almacenan todas las valoraciones que han realizado cada uno de los usuarios del sistema en relación a cada uno de los modelos de raqueta que forman parte de la base de datos. Cada celda( $i,j$ ) representa la puntuación que ha otorgado el usuario( $i$ ) a la raqueta( $j$ ).

Primero, se obtiene una lista de todos los usuarios que se han registrado en el sistema. Una vez hecho esto, se obtiene el número de usuarios y de raquetas que hay almacenados en el grafo de conocimiento para construir una tabla vacía (de momento) con las dimensiones adecuadas.

Después, se solicita a la base de datos una lista con todas las valoraciones que se han registrado para poder rellenar la tabla. Se recorre la lista y se van colocando

las puntuaciones en la posición que les corresponde según el usuario que valora y la raqueta que se puntúa. En las celdas que no corresponden a ninguna valoración, se mantiene el valor 0, ya que la puntuación mínima es un 1, de modo que no habrá confusión.

```
allRatings.forEach{  
    collabTable[user][racket] = rating  
}
```

### 6.1.2. Cálculo de similitud entre usuarios

Con la tabla ya completa, se puede realizar una serie de cálculos con el fin de obtener un valor que indique la similitud entre el usuario actual de la aplicación y todos los demás, a través de las opiniones almacenadas previamente.

Pese a la existencia de otras técnicas, como el coeficiente de correlación de Spearman, para este caso se ha empleado el de Pearson, no solo porque se vale de la comparación de las medias de las valoraciones, lo que resulta en un cálculo sencillo y computacionalmente eficiente, sino también porque existe una biblioteca para JavaScript, denominada jStat, que ya implementa esta técnica, lo que reduce los posibles errores que se puedan cometer en la implementación y ahorra tiempo en el desarrollo de la aplicación. En la **Fig. 3** se muestra la fórmula matemática a aplicar:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

**Fig. 12:** Fórmula de correlación de Pearson. Fuente [\[3\]](#)

Con la fórmula anterior, vamos a calcular la correlación entre el usuario actual  $x$  y todos los demás, siendo cada uno de estos últimos representados por  $y$ , teniendo en cuenta que  $y$  va a representar a un usuario diferente en cada iteración. Por tanto, tenemos:

$r$ : coeficiente de Pearson para el usuario actual  $x$  respecto al usuario  $y$

$x_i$ : valoración del usuario actual para la raqueta  $i$



$\bar{x}$ : media de las valoraciones realizadas por el usuario actual

$y_i$ : valoración del usuario  $y$  a la raqueta  $i$

$\bar{y}$ : media de las valoraciones del usuario  $y$

### 6.1.3. Predicción de puntuaciones

Una vez calculados todos los coeficientes de correlación se ordenan los usuarios según este valor respecto al usuario actual, de modo que las puntuaciones de los primeros usuarios de la lista serán las más parecidas al actual. De esta lista, se escogerán los 4 usuarios más parecidos.

Esta lista se utiliza para calcular las puntuaciones que el sistema predice que el usuario actual pondría a cada una de las raquetas que aún no ha valorado expresamente. La fórmula de predicción que se emplea es la siguiente:

$$\hat{r}_{uj} = \mu_u + \frac{\sum_{v \in P_u(j)} \text{Sim}(u, v) \cdot s_{vj}}{\sum_{v \in P_u(j)} |\text{Sim}(u, v)|} = \mu_u + \frac{\sum_{v \in P_u(j)} \text{Sim}(u, v) \cdot (r_{vj} - \mu_v)}{\sum_{v \in P_u(j)} |\text{Sim}(u, v)|}$$

**Fig. 13:** Fórmula de predicción de valoraciones.

### 6.1.4. Recomendación colaborativa

Las valoraciones obtenidas en el paso anterior, son las que se emplean, junto con las que el usuario actual ha realizado previamente, para decidir cuál de los modelos de raquetas presentes en la base de datos se adapta mejor a los gustos del susodicho.

## 6.2. Algoritmo de contenido

### 6.2.1. Selección de características

Para implementar un algoritmo de contenido satisfactorio, lo primero que hay que hacer es seleccionar cuidadosamente qué características son más decisivas a la hora de experimentar diferencias notorias en el juego entre unas raquetas y otras. Desde mi experiencia y la de otros jugadores que he tenido la oportunidad de consultar, esas características son: el peso, el área de la cabeza de la raqueta, la flexibilidad, el patrón de cuerdas y la marca. Además, a estas propiedades se le agrega el precio de la raqueta, dado que, aunque no afecta directamente al juego,

puede acotar la recomendación a nivel económico. Al estar la mayoría de estas características representadas por valores numéricos, se escogerán intervalos de valores para poder simplificar el problema y hacerlo computacionalmente más sencillo.

### 6.2.2. Creación de la tabla de contenido

Una vez seleccionado el vector de características, habrá que formar una tabla de contenido. En ella, las filas corresponden a cada una de las raquetas que forman parte de la base de datos, ordenadas según su "modelID" de forma ascendente. La primera fila estará formada por la valoración que le ha dado el usuario a la raqueta en cuestión. Luego, tendremos en las columnas todos los valores distintos que podamos encontrar para cada una de las propiedades anteriores. Por tanto, los colocaremos de la siguiente manera:

peso1 |...| pesoN | area1 |...| areaN |...| flex1 |...| flexN |...| patron1 |...| patronN |...| precio1 |...| precioN | marca1 |...| marcaN | valoracion

Al ser algunas de las características anteriores valores numéricos, se han escogido una serie de intervalos para poder acotar el filtrado y hacerlo más efectivo, dado que de esta manera será más sencillo encontrar similitudes entre raquetas. Se establecen intervalos para las siguientes propiedades:

Peso → Intervalos de 10g, entre 220g y 330g

Flexibilidad → Intervalos de 5 puntos, desde 45 hasta 75

Precios → Intervalos de 20\$, desde 80\$ hasta 240\$

De todas las propiedades a tener en cuenta, se le ha dado mayor importancia a las siguientes: peso de la raqueta (3 puntos), patrón de cuerdas (2 puntos) y precio (2 puntos). La función que genera la tabla de contenido es la siguiente (pseudocódigo):

```
function getContentTable(currentUser) {  
  
    getRacketProperties  
  
    setTableHeaders  
  
    getUserRatings(currentUser);  
  
    contentTable.rows.forEach(  
        fillRowWithRacketProperties
```

```
        addUserRating  
  
    }  
  
    return contentTable;  
}
```

Una vez obtenida la tabla de contenido, se busca una reducción de filas para mantener únicamente aquellas que corresponden a raquetas que el usuario actual ha valorado. En código se realiza de la siguiente manera:

```
contentTable.rows.forEach(  
    if(row == headers){  
        addRowToReducedTable  
    }else if(userHasRatedThisRacket){  
        addRowToReducedTable  
    }  
}
```

### 6.2.3. Generación del perfil de usuario

Con la tabla reducida, se podrá generar un perfil para el usuario de la aplicación. Para ello, se multiplicará el valor de cada una de las columnas de un modelo de raqueta por la valoración que el usuario le ha dado a esta, teniendo en cuenta que, al ser 0 aquellas celdas que corresponden a características no presentes en la raqueta, después de la multiplicación seguirán siendo 0.

Una vez hecho esto, se calculará la media para cada una de las columnas que forman parte de la tabla, sin tener en cuenta las celdas que tienen un 0, y se multiplicarán por la frecuencia con la que aparecen. Estos últimos pasos se definen en el programa de la siguiente manera (pseudocódigo):

```
function getUserProfile(table){  
  
    addHeadersToProfile
```

```
for j in [1, columns]:  
  
    for i in [1, rows]:  
  
        if (table[i][j]!=0):  
  
            columnAddition += table[i][j]  
  
            counter++  
  
    frecuencies.add(counter/rows)  
  
    average.add(columnAddition/counter)  
  
return profile
```

La generación del perfil no termina aquí. Para poder trabajar con él, habrá que normalizarlo, devolviendo cada valor al rango inicial, es decir, para aquellos que tienen un peso de 1, el rango será [0, 1], mientras que para los que tienen un peso diferente el rango será equivalente a [0, peso]. La normalización se realiza así:

```
function normalizeProfile(profile, frecuencies)  
  
    for i in [0, profile.columns]:  
  
        profile[1][i] = profile[1][i]/5*frecuencies[i];
```

#### 6.2.4. Recomendación por contenido

Con el perfil ya creado, se puede proceder a la recomendación. Para este paso, se emplea la fórmula de similitud de coseno, correspondiente a la **Fig. 4**:

$$\cos(\theta) = \frac{\sum_{i=1}^n (x_i \times y_i)}{\sqrt{\sum_{i=1}^n (x_i)^2} \times \sqrt{\sum_{i=1}^n (y_i)^2}}$$

**Fig. 14:** Fórmula de similitud de coseno. Fuente [\[4\]](#)

Esta fórmula se aplica a cada uno de los modelos de raquetas, para calcular la probabilidad de que le gusten al usuario. La aplicación de esta fórmula nos deja como resultado un vector de valores que pueden ser ordenados de mayor a menor para

obtener una lista de las raquetas que tienen una mayor probabilidad de adaptarse a las necesidades del usuario:

```
contentTable.rows.forEach{  
    if(notRatedYet):  
        getNumerator  
        getDenominator  
        similarityTable.add(numerator/denominator)  
}
```

De la lista obtenida se escogen los 3 modelos de raqueta para los que la similitud calculada ha sido mayor y se le muestran al usuario a través de la interfaz gráfica de la aplicación.

### 6.3. Algoritmo híbrido

El algoritmo híbrido combina los resultados del algoritmo de contenido con el colaborativo. Este será el que estará disponible para el usuario en una supuesta versión de producción del sistema.

Este algoritmo se divide en dos fases. La primera consiste en combinar el vector de resultados del colaborativo con el de contenido, de manera que ambos vectores contengan las mismas raquetas, aunque cada uno con una valoración propia. La segunda fase consiste en asignar un peso a cada uno de los algoritmos, de manera que la combinación no sea necesariamente equitativa, sino que pueda darse mayor relevancia a uno de ellos. En este caso, al contener el grafo un conjunto de usuarios simulados, se le ha otorgado un peso mayor al de contenido, aunque en una posible versión futura llevada a producción, esto podría variar.

### 6.4. Algoritmo de recomendación de cordajes

Tras obtener una recomendación de raquetas, al usuario se le muestra un cuadro en el que puede completar una serie de campos con el fin de conocer cuáles son los cordajes que más se adecúan a su estilo de juego y a sus gustos. En el cuadro se deben completar 4 campos:

1. De las raquetas recomendadas por el sistema, ¿cuál prefiere el usuario? Las 3 se muestran por orden de preferencia estimada, por lo que, en principio, la

mejor debería ser la primera, pero el usuario tiene poder de decisión para escoger una de las 3.

2. De los jugadores presentes en el sistema, ¿cuál es el que podría considerarse que tiene un juego más parecido al usuario?
3. Peso del usuario
4. Altura del usuario

El cuadro mostrado es el que se incluye en la **Fig. 15**:

## Get the string that best suits your racket and your game!

<b>1. Select one of the recommended rackets above</b>	<b>2. Select the player whose game is most similar to yours</b>
<div><input type="text" value="2"/>   v</div>	<div><input type="text" value="Andre Agassi"/>   v</div>
<b>3. Introduce your weight</b>	<b>4. Introduce your height</b>
<div>Weight (kg) <input type="text" value="70"/></div>	<div>Height (m) <input type="text" value="1.75"/></div>
<div>Get the recommended strings!</div>	

**Fig. 15:** Selección de características para recomendación de cordaje. Fuente propia

### 6.4.1. Base del algoritmo

Este algoritmo se basa en el de contenido. Sin embargo, tiene algunas modificaciones importantes. Evidentemente, la tabla de contenido para los cordajes es completamente distinta, debido a que las características a tener en cuenta son diferentes para los cordajes que para las raquetas. Con la tabla ya generada, los pasos posteriores son similares, ya que se reduce la tabla a los cordajes que hayan sido valorados por el usuario, se multiplican las filas por las valoraciones y se crea un perfil que será normalizado más tarde. Sin embargo, este algoritmo tiene un último paso adicional que se detalla a continuación.

### 6.4.2. Cálculo de bonus

Con el vector de valoraciones estimadas ya calculado, este se envía a una función que bonifica las valoraciones que cumplen algunos de los siguientes criterios:

#### 6.4.2.1. Cordaje usado por el jugador profesional seleccionado

Para cada cordaje del sistema, si el jugador que ha sido seleccionado por el usuario por tener características similares a él en el juego, su valoración se verá incrementada en 1 punto.

#### 6.4.2.2. Cordaje usado por un jugador de características físicas similares

Con base en el peso y la altura introducidas por el usuario, se calculará la distancia euclídea respecto a estas mismas características en referencia a los jugadores profesionales que forman parte del sistema. De esta manera, se escogerá a los jugadores cuya distancia respecto al usuario sea menor a un umbral establecido (0.5), por tener una complexión física similar. Aquellos cordajes que sean empleados por jugadores que se han determinado como similares, obtendrán un bonus de 0.5 puntos en la valoración estimada.

#### 6.4.2.3. Cordaje usado por algún jugador que emplea la raqueta elegida

Según la raqueta elegida, se obtiene una lista de jugadores que la utilizan. A partir de esta lista, se obtiene otra que incluye los cordajes que estos emplean con esa raqueta. De esta manera, se sumarán 0.5 puntos a aquellos cordajes que son empleados por algún jugador cuya raqueta es la seleccionada por el usuario.

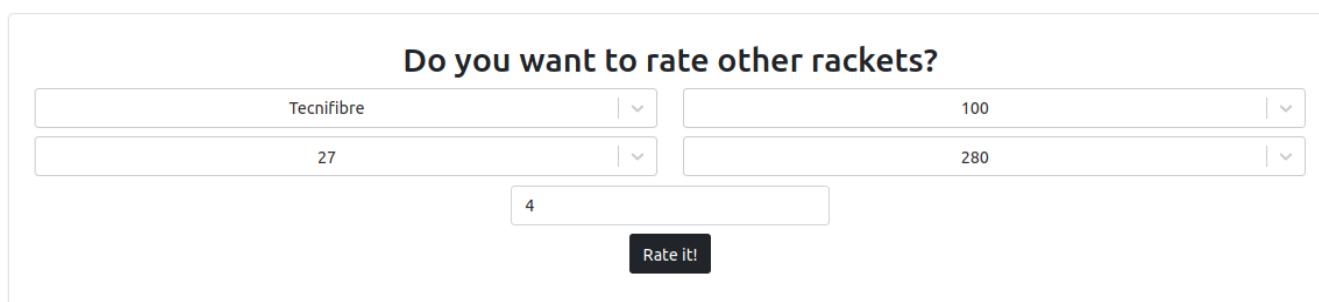
## 7. ANÁLISIS DE RESULTADOS

Para realizar un análisis de resultados de los algoritmos del sistema, veremos dos casos de uso completamente diferentes. Se analizarán tanto el algoritmo de contenido como el colaborativo. Para cada uno de ellos, se propondrá el caso de que use la aplicación un niño de unos 12 años con una masa corporal de 50 kg y una altura de 1.55 m o un hombre adulto de 30 años que mida 1.82 m y pese 80 kg.

## 7.1. Caso 1: Niño

Para este caso, se ha creado un usuario 'child1', con contraseña 'Child1!!', que tiene las características mencionadas anteriormente. Las raquetas que este niño valora positivamente son, principalmente, aquellas más manejables, es decir las que tienen un peso ligero. Además, se encontrará más cómodo con raquetas que tengan un patrón de cuerdas menor (por ejemplo, 16x19), porque sentirá que la raqueta despide la pelota con más fuerza. Por último, también suponemos que un usuario de esas características preferirá una raqueta con un tamaño de cabeza amplio, que permita mayor margen de error en el golpeo.

Siguiendo estas pautas, utilizando el puntuador de la aplicación (mostrado en la **Fig. 16**) , se incluyen en el grafo de conocimiento las valoraciones indicadas en la tabla de la **Fig. 17**.



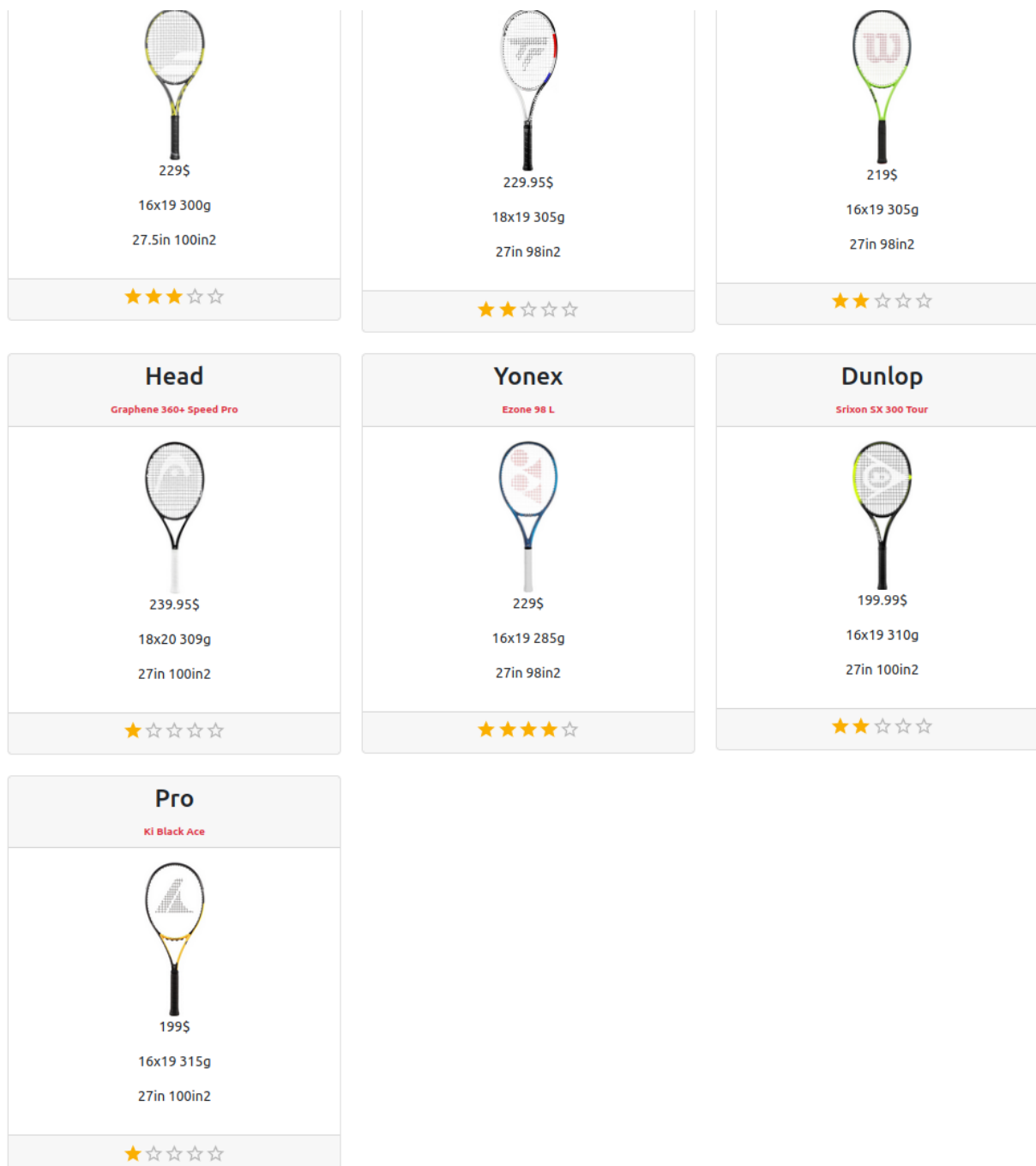
**Fig. 16:** Selección de características para puntuación de raquetas. Fuente propia

modelID	brand	price_Dol	headSize_in2	length_in	weightUnstrung_g	minTension_kg	maxTension_kg	beamWidth_mm	flex	powerLevel	stringPattern_VxH	RATING
2	Tecnifibre	209.95	100	27	280	22.22	24.94	23	70	Medium	16x19	4
6	Wilson	129	100	27	261	22.67	27.21	23	66	Medium	18x16	4
16	Head	189.95	104	27	270	23.58	28.12	24	60	High	16x20	4
79	Dunlop	179.99	100	27	270	20.41	29.48	23	67	Medium	16x19	4
84	Head	189.95	105	27	265	23.58	28.12	23	66	Medium	16x19	5

**Fig. 17:** Valoraciones realizadas usando el puntuador. Fuente propia

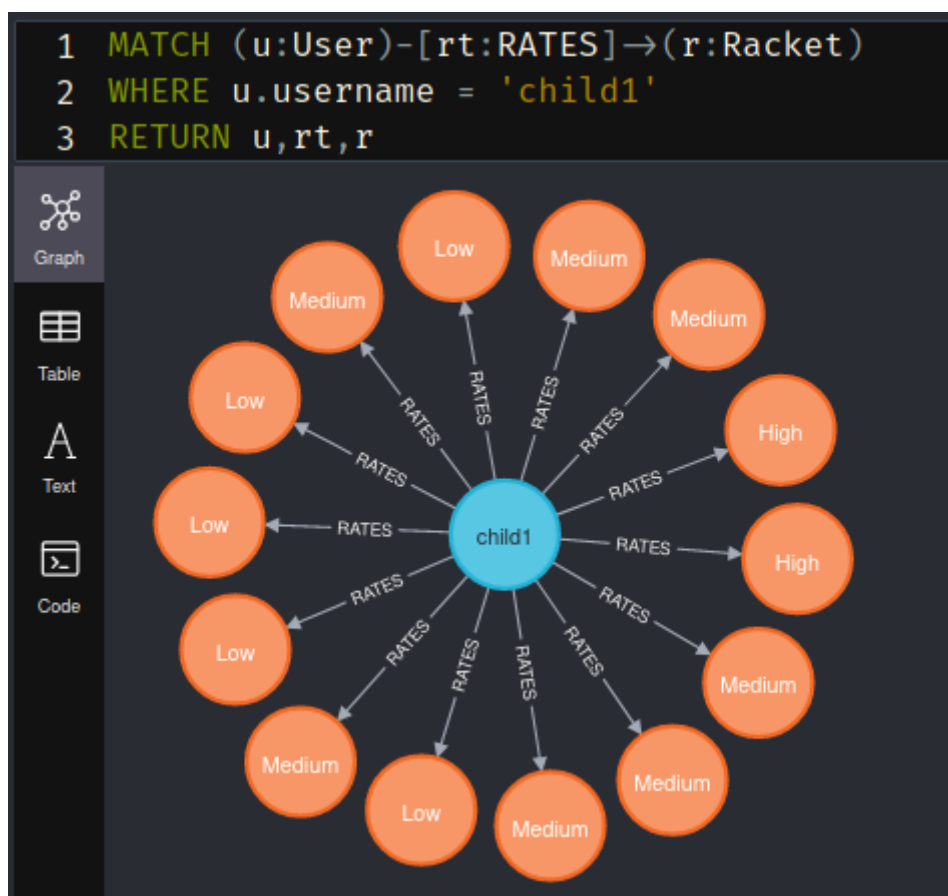
De nuevo siguiendo la misma línea, se procede a puntuar las raquetas que se ofrecen por defecto en la página principal, utilizando para ello la interfaz de valoración que se muestra asociada a cada raqueta, como se puede ver en la **Fig. 18**.





**Fig. 18:** Valoraciones realizadas a las raquetas por defecto para niño. Fuente propia

Tras estas valoraciones, el grafo de conocimiento incluye las relaciones mostradas en la **Fig. 19**, que es una captura del gestor de Aura para la base de datos.






**Fig. 19:** Valoraciones mostradas como relaciones en el grafo. Fuente propia

### 7.1.1. Algoritmo colaborativo

Una vez realizadas las valoraciones pertinentes, se pide al sistema que ejecute el algoritmo colaborativo, utilizando para ello, aparte de las valoraciones realizadas para el usuario actual, todas aquellas almacenadas previamente en la base de datos, correspondientes a los usuarios generados aleatoriamente mediante un script en Python.

En la **Fig. 20** se muestran los resultados obtenidos. Efectivamente, podemos comprobar que las raquetas recomendadas cumplen a la perfección con los requisitos comentados en la introducción de esta sección. Obtenemos 3 raquetas, todas ellas con un patrón de cuerdas de tipo 16x19, ligeras y amplias. Por tanto, se puede concluir que la ejecución ha sido un éxito para este caso de uso.

### Recommended Rackets




Racket 1	Racket 2	Racket 3
		
<b>Wilson</b>	<b>Head</b>	<b>Babolat</b>
139\$	179.95\$	199\$
16x19 278g	16x19 270g	16x19 269g
27in 100in2	27.2in 107in2	27in 100in2

**Fig. 20:** Resultados del algoritmo colaborativo para niño. Fuente propia

### 7.1.2. Algoritmo de contenido

Con la misma información en el grafo de conocimiento que para el apartado anterior, se ejecuta el algoritmo de contenido y se obtiene el resultado mostrado en la **Fig. 21**. Como se puede apreciar, una de las raquetas recomendadas coincide con las anteriores. Las otras dos siguen cumpliendo el mismo patrón. Son raquetas ligeras, con cabeza amplia y patrón de cordaje cómodo (16x19). De nuevo, el resultado es exitoso.

### Recommended Rackets

Racket 1	Racket 2	Racket 3
		
<b>Wilson</b>	<b>Volk</b>	<b>Head</b>
139\$	189.99\$	189.95\$
16x19 278g	16x19 275g	16x19 275g
27in 100in2	27in 100in2	27in 105in2

**Fig. 21:** Resultados del algoritmo de contenido para niño. Fuente propia

### 7.1.3. Algoritmo de recomendación de cordajes

En este caso, no procede emplear el algoritmo de recomendación de cordajes, porque está parcialmente basado en el físico y el estilo de juego del usuario en comparación con jugadores profesionales. El físico de un jugador profesional nunca va a ser similar al de un niño de las características indicadas previamente, por lo que este algoritmo quedará limitado a usuarios de edad adulta o, al menos, de características físicas propias de un adulto.

## 7.2. Caso 2: Adulto

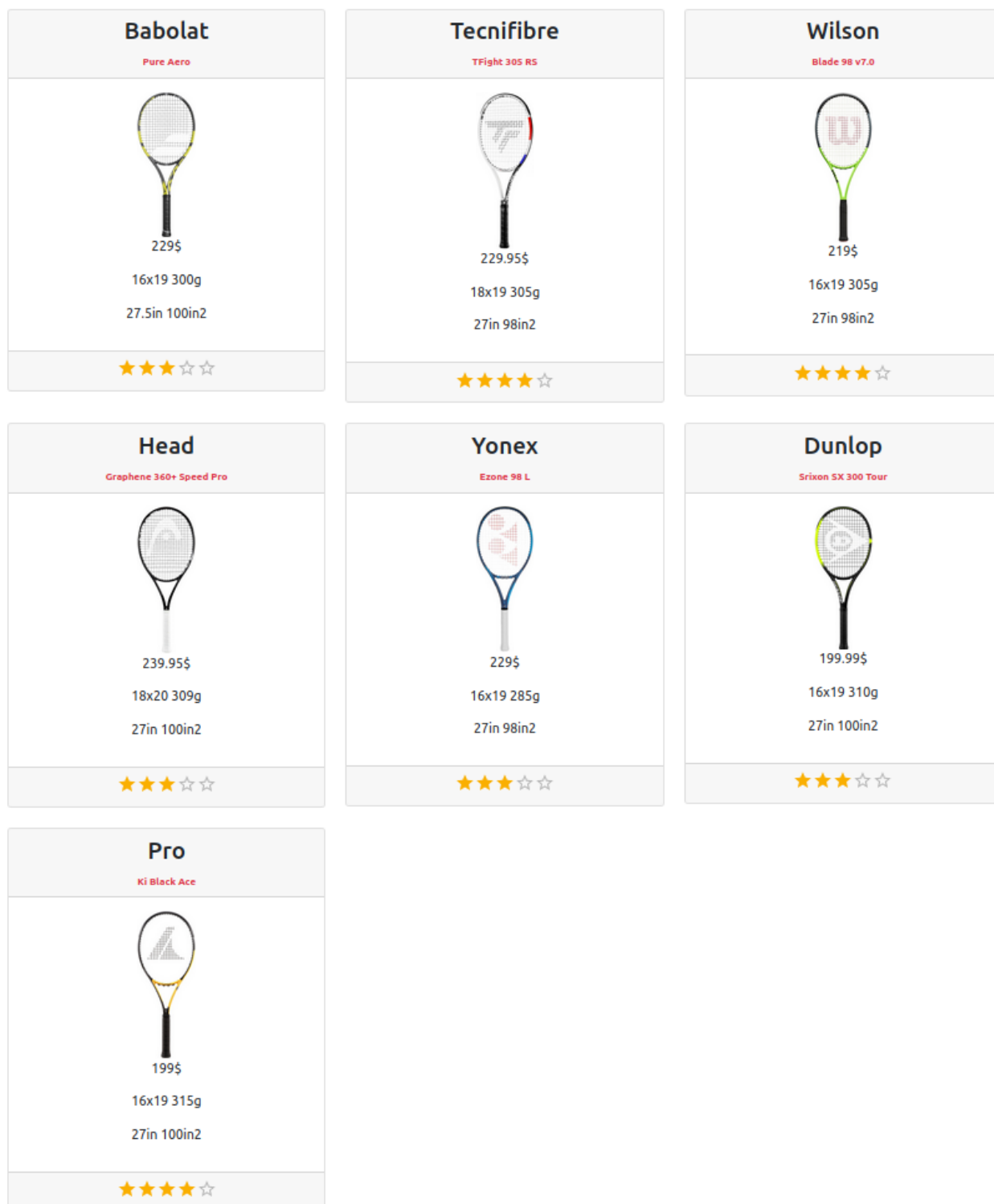
Para este segundo caso, se ha creado un usuario 'adult1', con contraseña 'Adult1!!', que tiene las características definidas en la introducción de la sección 7 (unos 30 años, 1.82 m de altura y 80 kg de peso). Este jugador estará capacitado para utilizar raquetas más pesadas, que le transfieran mayor consistencia y potencia al golpeo. Además, si suponemos que su nivel es avanzado, se encontrará a gusto con raquetas con un punto dulce más reducido (menor área de la cabeza) y, probablemente, estará dispuesto a desembolsar una cantidad mayor de dinero. El patrón de cuerdas, en este caso, dependerá de sus gustos personales.

Siguiendo estas pautas, utilizando el puntuador de la aplicación (mostrado en la **Fig. 16**), se incluyen en el grafo de conocimiento las valoraciones indicadas en la tabla de la **Fig. 22**.

modelID	brand	price_Dol	headSize_in2	length_in	weightUnstrung_g	minTension_kg	maxTension_kg	beamWidth_mm	flex	powerLevel	stringPattern_VxH	RATING
1	Head	239.95	98	27	315	21.77	25.85	20	64	Low	16x19	5
22	Yonex	239	98	27	305	20.41	27.21	22.5	65	Low	16x19	4
36	Babolat	229	100	27	315	22.67	26.76	23	72	Low	16x19	4
52	Volkl	209.99	100	27	310	22.67	27.21	24	68	Low	16x19	4
54	Tecnifibre	229.95	98	27	315	22.22	26.76	21.7	66	Low	18x20	5
133	Wilson	219	98	27	305	22.67	27.21	20.6	62	Low	18x20	4

**Fig. 22:** Valoraciones realizadas usando el puntuador. Fuente propia

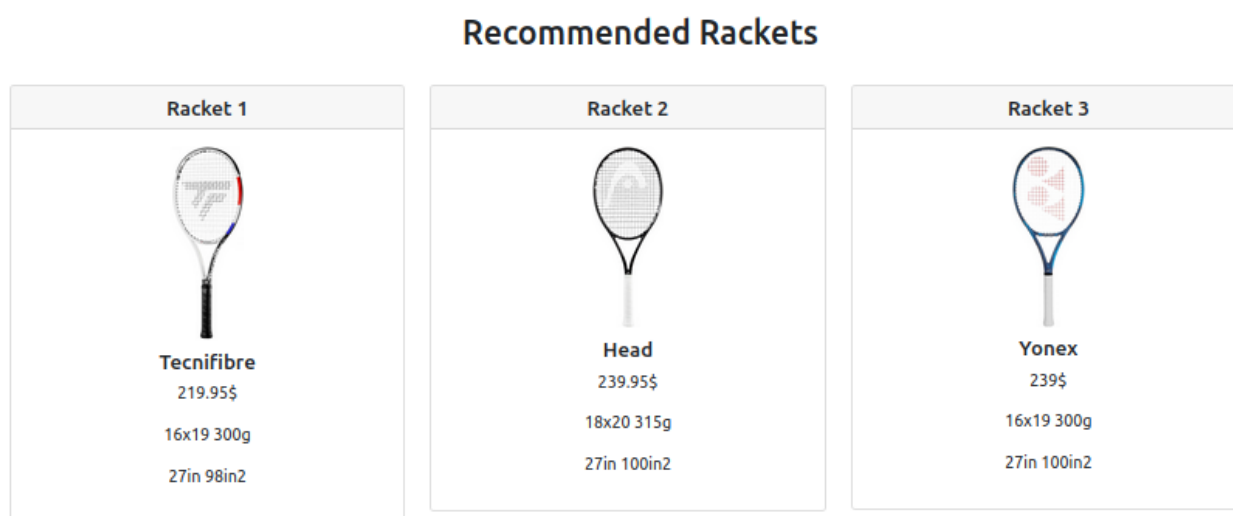
De nuevo siguiendo la misma línea de gustos y estilo de juego, se procede a puntuar las raquetas que se ofrecen por defecto en la página principal, utilizando para ello la interfaz de valoración de la **Fig. 23**, igual que en el caso del niño.



**Fig. 23:** Valoraciones realizadas a las raquetas por defecto para adulto. Fuente propia

### 7.2.1. Algoritmo colaborativo

Una vez valoradas las raquetas anteriores, se ejecuta el algoritmo colaborativo, obteniéndose los resultados de la **Fig. 24**. En ella, vemos que, como se esperaba, las raquetas que nos recomienda el sistema cumplen las características expuestas en un principio, es decir, tenemos una lista de raquetas con un cierto peso (no manejables para cualquier jugador), con un punto dulce reducido y un patrón de cuerdas que no es constante por no haberlo definido en un principio. Además, los modelos que se muestran tienen precios elevados. Se podría haber encontrado un par de raquetas con un peso mayor de 300 g, pero, en general, el algoritmo ha encontrado raquetas adecuadas al usuario.



**Fig. 24:** Resultados del algoritmo colaborativo para adulto. Fuente propia

### 7.2.2. Algoritmo de contenido

Con la mismas valoraciones realizadas que para el caso colaborativo, como resultado del algoritmo de contenido tenemos la lista de la **Fig.25**. Quizá, en este caso, podríamos decir que el resultado de contenido ha sido aún más preciso que el colaborativo, dado que el peso de las raquetas obtenidas es mayor y el tamaño de la cabeza, menor. Además, el precio es elevado al igual que el algoritmo anterior y el patrón de cuerdas es el estándar (16x19) para todos los modelos. Este resultado ha sido, por tanto, exitoso.

### Recommended Rackets










Racket 1	Racket 2	Racket 3
		
<b>Head</b>	<b>Yonex</b>	<b>Yonex</b>
229.95\$	239\$	229\$
16x19 315g	16x19 310g	16x19 315g
27in 95in2	27in 97in2	27in 98in2

**Fig. 25:** Resultados del algoritmo de contenido para adulto. Fuente propia

#### 7.2.3. Algoritmo de recomendación de cordajes

En este caso sí que procede emplear el algoritmo de recomendación de cordajes, puesto que el usuario sí que tiene las características físicas y de juego adecuadas para ser comparado con jugadores profesionales. Su peso y estatura puede ser similar y, el jugador puede tener un estilo ya definido que se asemeje al de algún profesional.

Para este ejemplo, supondremos que el usuario tiene un juego similar a Richard Gasquet, un jugador defensivo con un potente revés a una mano. La potencia de estos tiros requiere un cordaje grueso que soporte fuertes impactos de la pelota. Además, el cordaje será preferiblemente monofilamento (más resistente). Con esta información, el usuario realizará una valoración similar a la de la **Fig. 26**. Previsiblemente, puntuará positivamente que el cordaje sea monofilamento o híbrido y negativamente que sea multifilamento o de tripa natural. Además, los cordajes mejor valorados serán los que tienen un grosor de, al menos, 1.25 mm.

<p><b>Wilson</b></p> <p>Luxilon Big Banger ALU Power 16</p>  <p>Monofilament</p> <p>Aluminium</p> <p>1.25mm</p> <p>★★★★☆</p>	<p><b>Solinco</b></p> <p>Hyper-G 19</p>  <p>Monofilament</p> <p>Co-polyester</p> <p>1.1mm</p> <p>★★★☆☆</p>	<p><b>Prince</b></p> <p>Synthetic Gut Dura Flex 16</p>  <p>Multifilament</p> <p>Nylon</p> <p>1.3mm</p> <p>★★★★☆☆</p>
<p><b>Babolat</b></p> <p>VS Touch 16</p>  <p>Gut</p> <p>Gut</p> <p>1.3mm</p> <p>★★★★☆☆</p>	<p><b>Yonex</b></p> <p>Poly Tour Pro 17</p>  <p>Monofilament</p> <p>Polyester</p> <p>1.2mm</p> <p>★★★★☆☆</p>	<p><b>Tecnifibre</b></p> <p>NRG2 17</p>  <p>Multifilament</p> <p>Elastyl Fiber</p> <p>1.24mm</p> <p>★★★☆☆</p>
<p><b>Head</b></p> <p>Rip Control 17</p>  <p>Multifilament</p> <p>Polyolefin</p> <p>1.25mm</p> <p>★★★★☆☆</p>	<p><b>Wilson</b></p> <p>Luxilon M2 Pro 16L</p>  <p>Multifilament</p> <p>Polyester</p> <p>1.25mm</p> <p>★★★★☆☆</p>	<p><b>Wilson</b></p> <p>Luxilon Big Banger ALU Power 17</p>  <p>Monofilament</p> <p>Aluminium</p> <p>1.2mm</p> <p>★★★★☆☆</p>

**Fig. 26:** Cordajes valorados por adulto de nivel avanzado. Fuente propia

Para obtener la recomendación de cordajes, hay que pasar antes por la de raquetas. En este caso, emplearemos la recomendación de contenido, así que las raquetas a elegir son las mostradas en la **Fig. 25**. Una vez el usuario ha llegado a este punto, ya puede introducir sus características físicas, la raqueta seleccionada de las 3 anteriores y el jugador cuyo estilo de juego se asemeja en mayor medida al suyo propio. Los campos se rellenarían de la manera que se observa en la **Fig. 27** y la recomendación obtenida es la de la **Fig. 28**. En ella, queda patente que se premia por encima de todo el grosor del cordaje y el tipo (monofilamento o híbrido), como se



esperaba en un principio. Las tres recomendaciones tienen composiciones diferentes, pero no hemos dado importancia a este factor, en un principio.

**Get the string that best suits your racket and your game!**

1. Select one of the recommended rackets above

2. Select the player whose game is most similar to yours




3. Introduce your weight

4. Introduce your height

Get the recommended strings!

**Fig. 26:** Cordajes valorados por adulto de nivel avanzado. Fuente propia

**Recommended Strings**

String 1	String 2	String 3
 <b>Wilson</b> Polyester Hybrid 1.32mm	 <b>Wilson</b> Co-polyester Monofilament 1.3mm	 <b>Wilson</b> Co-polymer Monofilament 1.3mm

**Fig. 27:** Cordajes recomendados para adulto de nivel avanzado. Fuente propia

## 8. ANÁLISIS CRÍTICO DEL SISTEMA

### 8.1. Beneficios

Una de las ventajas del sistema es la propia temática, dado que las raquetas, aunque se van renovando y cada año aparecen nuevos modelos, la tecnología no varía

en gran medida y, generalmente, los modelos nuevos solo cambian la estética u otros detalles menores respecto a los modelos de años anteriores.

Además, la interfaz gráfica es bastante amigable e intuitiva, en general, pues se usan elementos muy conocidos como la valoración a través de iconos estrellados o el login diseñado por Auth0. Este login también permite el almacenamiento de claves de manera segura e, incluso, permite la sincronización con Google para que el proceso de inicio de sesión se lleve a cabo en su propia plataforma, de manera que la clave no tiene que ser expuesta en ningún momento.

En cuanto a la propia recomendación, vemos que para los dos casos de estudio anteriores, los algoritmos dan resultados lógicos y coherentes con las valoraciones y otros datos aportados por el usuario de la aplicación.

## 8.2. Limitaciones

En cuanto a las limitaciones del sistema, cabe destacar el hecho de que, actualmente, el conjunto de usuarios incluidos en la base de datos es un conjunto simulado, de manera que, en el caso de llevarse a producción habría que escoger a un grupo de jugadores de tenis de un nivel amateur, es decir, dentro del público objetivo de la aplicación, para que aportase valoraciones reales y fundamentadas, con el objetivo de adquirir un nivel de precisión mucho más alto. De esta manera, se podría conseguir que el algoritmo colaborativo generase recomendaciones con un grado mucho más alto de fiabilidad.

Por otra parte, como se explicaba en la sección 7, el algoritmo de recomendación de cordajes no es aplicable a usuarios que no tengan un físico propio de un adulto, por no estar presentes en la base de datos jugadores conocidos en su etapa de menores, sino ya en el circuito profesional. En la sección de líneas de futuro se detalla esta cuestión.

## 8.3. DAFO

Las ventajas y limitaciones anteriores se pueden mostrar en forma de gráfico DAFO (Debilidades, Amenazas, Fortalezas y Oportunidades). Este tipo de análisis permite estudiar la situación del proyecto BreakPoint, interna y externamente.

En la tabla siguiente se incluyen cada una de las características a comentar dentro de esta sección:

	INTERNO	EXTERNO
NEGATIVO	<b>Debilidades</b> <ul style="list-style-type: none"><li>• Algoritmos sencillos</li><li>• Datasets dispares</li></ul>	<b>Amenazas</b> <ul style="list-style-type: none"><li>• Ataque al servidor</li><li>• Generación de usuarios ficticios</li></ul>
POSITIVO	<b>Fortalezas</b> <ul style="list-style-type: none"><li>• Sencillez de implementación</li><li>• Eficacia en la recomendación</li></ul>	<b>Oportunidades</b> <ul style="list-style-type: none"><li>• Aplicación en empresas</li><li>• Posibilidad de expansión</li></ul>

## 9. LÍNEAS DE FUTURO

Una vez desarrollado el sistema recomendador, podría servir como una orientación a un jugador de tenis, pero para ello, el sistema tendría que haber sido "alimentado" previamente con valoraciones reales y fundamentadas, dado que, actualmente, el conjunto de usuarios almacenados en la base de datos del sistema es ficticia, aunque las valoraciones realizadas se han agregado de acuerdo a un criterio (no aleatorio). Si estas valoraciones llegasen a ser reales y abundantes, se podría modificar los pesos del algoritmo híbrido, dado que, por las razones anteriores, se le da mayor peso al algoritmo de contenido que al colaborativo.

Una aplicación de estas características podría ser empleada por una tienda de deportes que tenga una sección de tenis o, incluso, por una que esté enteramente especializada en ello. En el caso de comercios "on-line", al no poder los clientes ser aconsejados por un dependiente, ni tampoco probar ninguno de los modelos a la venta, la implantación de este sistema dentro de los servicios que ofrece la página sería una muy buena forma de suplir estas carencias. Además, promovería la interacción con los clientes y se podría contratar a expertos que realicen tests a las raquetas, cuya valoración podría ser considerada de mayor peso a la hora de realizar una recomendación a un usuario.

Una de las posibles mejoras, tendría que ver con ser capaz de analizar la forma de golpear a la pelota que tiene el cliente. Si hubiese una forma factible de hacerlo, eso podría ayudar mucho a la recomendación, ya que se conocen los datos del golpeo que realizan los jugadores profesionales. De hecho, muchos de estos forman parte del grafo de conocimiento, de manera que sería muy sencillo utilizarlos para la recomendación.

Otra posible mejora, sería ampliar el "dataset" de jugadores profesionales, añadiendo datos de mujeres. Además, se podría recabar información acerca de jugadores jóvenes, que disputan competiciones nacionales e internacionales, para que el sistema sea también efectivo para usuarios con características físicas similares a ellos.

## 10. CONCLUSIONES Y APRENDIZAJE

Después de todas las horas dedicadas a este proyecto, puedo decir que estoy satisfecho con el trabajo realizado y me enorgullece comprobar que los conocimientos que he adquirido son amplios y variados, además de útiles para mi futuro profesional.

Las destrezas adquiridas comprenden, no solo conocimientos sobre la creación y manejo de bases de datos de grafos con Neo4j, sino también sobre el desarrollo de aplicaciones web con React.js. Además, he descubierto servicios como Auth0 que facilitan el desarrollo de aplicaciones de este tipo. Aparte de esto, la bibliografía relacionada con algoritmos de recomendación me ha resultado de gran ayuda para el proyecto y no descarto recurrir a ella en un futuro no muy lejano.

Por último, cabe destacar que la aplicación desarrollada funciona de manera eficaz, aunque quizás no de la forma más eficiente posible (posible mejora para el futuro). Esto queda patente en los análisis realizados en la sección 7, que demuestran que los modelos de raquetas y cordajes recomendados a cada tipo de usuario son los adecuados.

## 11. BIBLIOGRAFÍA Y OTRAS FUENTES

[1] Dan Arias (2020). [The Complete Guide to React User Authentication With Auth0](#). Consultado (2021, 15, noviembre)

[2] Leoyuanlou. [Tennis Racquets specs](#). Consultado (2021, 15, diciembre)

[3] Analytics Vidhya (2021, 6, enero). [Beginner's Guide to Pearson's Correlation Coefficient](#). Consultado (2021, 4, diciembre)

[4] Programador clic (2015). [Algoritmo de similitud - Similitud de coseno](#). Consultado (2021, 1, diciembre)

[5] Neo4j (2021). [Neo4j Browser for Aura](#).

[6] Neo4j (2021). [Neo4j](#).



[7] React.js (2021). [React.js](#).

[8] Auth0 (2021). [Auth0](#)

[9] Amy Hodler and Mark Needham. [Graph Data Science \(GDS\) For Dummies®](#). Consultado (2021, 12, diciembre)

[10] Juan Sequeda and Ora Lassila. [Designing and Building Enterprise Knowledge Graphs](#). Consultado (2021, 15, diciembre)