



universidad
de león



Escuela de Ingenierías Industrial, Informática y Aeroespacial

GRADO EN INGENIERÍA INFORMÁTICA

Sistemas de Información de Gestión y Business Intelligence

Memoria de la aplicación: YourBestCar

Sergio Rodríguez Valdeón

07 de noviembre de 2020

Índice

1. DESCRIPCIÓN DEL PROBLEMA	3
2. HERRAMIENTAS.....	4
3. APLICACIÓN	6
3.1 BASE DE DATOS	6
3.2 FRONTEND	9
3.3 BACKEND	12
4. ALGORITMO DE RECOMENDACIÓN	14
4.1 ALGORITMO DE RECOMENDACIÓN POR EDAD	14
4.2 ALGORITMO DE RECOMENDACIÓN COLABORATIVO.	15
5. ANÁLISIS DE RESULTADOS	18
5.1 EJEMPLO 1	18
5.2 EJEMPLO 2	19
5.3 EJEMPLO 3	21
5.4 EJEMPLO 4	23
6. DAFO	26
7. LÍNEAS DE FUTURO	27
8. LECCIONES APRENDIDAS	29
9. BIBLIOGRAFÍA.....	30

1.Descripción del problema

Para muchas familias, cuando llega la hora de cambiar de coche, encontrar uno que cubra todas sus necesidades se convierte en una odisea, ya que son bienes de gran valor y compras que se realizan cada muchos años hay tomar correctamente la decisión de cual elegir. Para ello normalmente todas las familias deciden invertir unos días en visitar todos los concesionarios de su ciudad, o incluso de alrededores, probando infinidad de coches. Una vez probados todos comparar los que más se adapten a tus necesidades y gustos, comparando: valor, motor, tipo de tracción, numero de puertas... A esto se le añade el preguntar a conocidos su opinión, o buscar en foros información acerca de los vehículos, si dan problemas, coste de los mantenimientos...

Por esto he decidido crear un sistema de recomendación de vehículos, que reduzca el tiempo de búsqueda y el primer filtrado, para facilitar la vida del usuario, y que pierda el menor tiempo posible, de esta manera el usuario solo tendrá que ir a los concesionarios donde de verdad están los coches que le interesan.

En el documento expuesto a continuación vamos a realizar una descripción completa sobre el desarrollo y funcionamiento del sistema de recomendación.

2.Herramientas

Lo primero que debemos comentar es que es un sistema de recomendación, como su nombre indica es una importante herramienta que establece un conjunto de recomendaciones de usuarios sobre unos datos para realizar predicción sobre recomendaciones a usuarios que le puedan ser de utilidad y así mejorar su experiencia de usuario. Estos sistemas suelen seguir una temática, en mi caso coches, pero tenemos ejemplo como música (Spotify), películas y series (Netflix) o video (YouTube).

Para este sistema vamos a utilizar un sistema de recomendación basado en una base de datos orientada a grafos. Para implementar esta base he decido usar NEO4J.

NEO4J es un servicio implementado en Java, el cual usa grafos para representar datos y relaciones entre ellos, se utilizan círculos para representar los nodos y aristas para representar las relaciones entre ellos, esto nos permite extraer valores con un gran rendimiento, de forma ágil, flexible y altamente escalable.

Para mí era una tecnología nueva, pero Neo4j ofrece gran cantidad de documentación, ya sea en forma de video o escrita. Además, realicé el curso inicial que te permite obtener un conocimiento completo del lenguaje y las funcionalidades para realizar operaciones de nivel medio.

Para crear esta base he decidido descargar un csv de Kaggle, esta página contiene gran cantidad de dataset de distintos tipos y con distintas características. Después de explorar un poco me quede con uno que, aunque tenía información que no necesitaba me pareció el más completo. En el apartado siguiente se explicará la normalización del dataset y la creación de la base.

Decidí implementar el sistema en una aplicación web responsive, de esta manera permitir el acceso desde cualquier dispositivo. Estas aplicaciones constan de 2 partes, el Frontend, que sería la vista de la aplicación propiamente dicha, y el backend que es el que se comunica entre la base de datos y el Frontend.

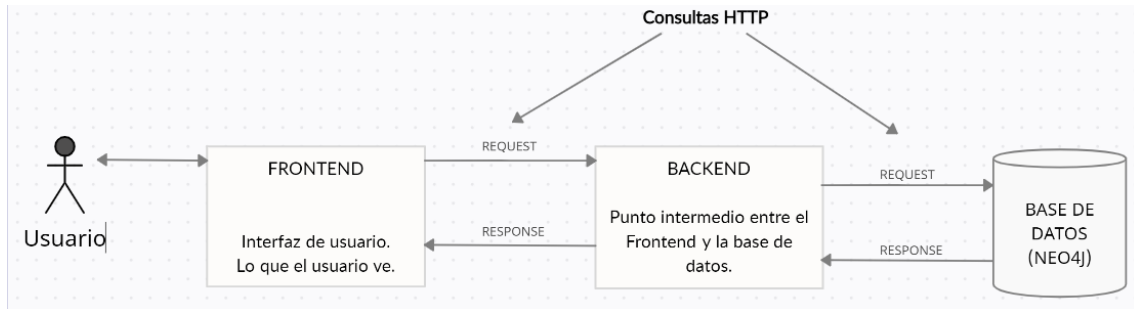
Para crear el Frontend he decidido utilizar Vuetify, el cual es un framework que combina el popular framework para realizar SPA, VueJS con la estética de Material Design. Esto me permite desarrollar una aplicación limpia y simple de manera sencilla.

En cambio, para el backend tuve dudas entre realizarlo en Python o JavaScript, ya que Python te ofrece mayor potencia de cálculo, pero no tengo conocimiento para utilizarlo en un sistema de tipo, por lo que tuve que optar por JavaScript ya que aprender Python me llevaría demasiado tiempo. Para ello utilice NodeJs que es un entorno de tiempo de ejecución de Javascript, y te permite ejecutar los js con un comando. Además, para realizar la comunicación

entre el backend y la base se utilizó el neo4j-driver, un módulo que ya viene implementado en JavaScript y habilita la comunicación. Y se utilizó Express que permite la comunicación entre Backend y Frontend.

3. Aplicación

Como hemos explicado en el apartado anterior la estructura básica de la misma sería:



Su funcionamiento básico sería, por ejemplo, el usuario quiere obtener un listado de todos los vehículos, se envía esta petición desde el Frontend al backend, y este a su vez realiza la petición a la base, en este caso lenguaje Cypher, la respuesta realiza el camino inverso y el Frontend se encarga en mostrarlos, en forma de lista, tabla, solo el nombre...

Este tipo de peticiones se realizan en un breve espacio de tiempo, menos de 1 segundo, lo que más puede retrasar la petición es una consulta a la base de datos que se demore más de lo debido.

Una vez sabido el funcionamiento, vamos a dividirlo en 3 grandes bloques: Base de datos, Backend y Frontend.

3.1 Base de datos

Para crear la base, partimos del csv, contaba con 11914 registros y las siguientes columnas:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O			
Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven	Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highway MPG	city mpg	Popularity	MSRP	
BMW	1 Series M	2011	premium unleaded (required)	335	6	MANUAL	rear wheel drive	2	"Factory Tuner	Luxury	High-Performance"	Compact	Coupe	26	19	3916	46135
BMW	1 Series	2011	premium unleaded (required)	300	6	MANUAL	rear wheel drive	2	"Luxury	Performance"	Compact	Convertible	28	19	3916	40650	

Lo primero que vi es que había valores N/A y 0, por lo que procedí a borrar las líneas con valores defectuosos, esto lo hice realizando una conversión a .xlsx y usando la funcionen de búsqueda. Después de una valoración decidí eliminar las columnas: Engine Fuel Type, Engine CylindersMarket Category,Vehicle Style y Popularity, también se formateo el nombre de las columnas para eliminar espacios y que no hubiese problemas a la hora de leer datos.

Comencé creando un nodo centra llamado Cars, continué creando un nodo Brand con el nombre de cada marca y relacionándolos entre sí con la relación MARCA.

```
LOAD CSV WITH HEADERS FROM "file:///data.csv" AS line
Merge (:Brand { name: line.Make})
Create(:Cars)
MATCH (a:Cars),(b:Brand)
CREATE (a)-[r:MARCA]->(b)
```

A continuación, se importaron todos los vehículos con las propiedades necesarias:

```
LOAD CSV WITH HEADERS FROM "file:///data1.csv" AS line

Create (:Car { model: line.Model,marca:line.Make,transmision: line.Trasmission, traccion:line.Driven_Wheels, puertas:line.Number_Doors, tamaño:line.Vehicle_Size, year: line.Year, cv: line.Engine_HP,consumo_urbano: line.city_mpg, consumo_extraurbano: line.highway_MPG, precio:line.MSRP})
```

Tuve que normalizar varios datos ya que el dataset no estaba correcto del todo, se muestran a continuación:

- Propiedad “tracción”: existía el valor “four Wheel drive” y “all wheel drive”, por lo que se modificó y se sustituyeron el primer valor por el segundo en todos los nodos.

```
MATCH (n:Car {traccion:"four wheel drive"}) set n.traccion="all wheel drive"
```

- Propiedad Transmision: se cambiaron los valores de “UNKNOWN” por “MANUAL”; “DIRECT_DRIVE” Y “AUTOMATED_MANUAL” por “AUTOMATIC”

```
MATCH (n:Car {transmision:"UNKNOWN"}) set n.transmision="MANUAL"
MATCH (n:Car {transmision:"DIRECT_DRIVE"}) set n.transmision="AUTOMATIC"
MATCH (n:Car {transmision:"AUTOMATED_MANUAL"}) set n.transmision="AUTOMATIC"
```

Cree los nodos para las propiedades, esto ayudaría en un futuro a mejorar la eficacia de las recomendaciones. Estos fueron:

3 de tipo Tracción para los 3 valores posibles:

```
create(:Traccion {name:"all wheel drive"})
create(:Traccion {name:"front wheel drive"})
create(:Traccion {name:"rear wheel drive"})
```

2 de tipo Puerta para cada valor (2,4):

```
create(:Puerta {numero:4})
create(:Puerta {numero:2})
```

2 de tipo Transmision para cada valor(AUTOMATIC, MANUAL):

```
CREATE (:Transmision {name:"AUTOMATIC"})
CREATE (:Transmision {name:"MANUAL"})
```

3 de tipo Tamaño para cada valor (Compact,Midsize,Large):

```
create(:Tamaño {name:"Compact"})
create(:Tamaño {name:"Midsize"})
create(:Tamaño {name:"Large"})
```

Posteriormente, relacione cada vehículo con el nodo el cual poseía su propiedad de la siguiente manera:

```
MATCH (b:Brand),(c:Car)
WHERE c.brand = b.name
CREATE (b)-[r:FABRICA]->(c)

MATCH (p:Puerta),(c:Car)
WHERE toInteger(c.puertas) = p.numero
CREATE (c)-[r:TIENE_PUERTAS]->(p)

MATCH (t:Traccion),(c:Car)
WHERE c.traccion= t.name
CREATE (c)-[r:TRACCION_EN]->(t)

MATCH (t:Transmision),(c:Car)
WHERE c.transmision= t.name
CREATE (c)-[r:TRANSMISION_IS]->(t)

MATCH (t:tamaño),(c:Car)
WHERE c.tamaño= t.name
CREATE (c)-[r:TAMAÑO_IS]->(t)
```

Y una vez relacionados, elimine las propiedades para que no hubiese información repetida y fallos de inconsistencia:

```
MATCH (c:Car )
```

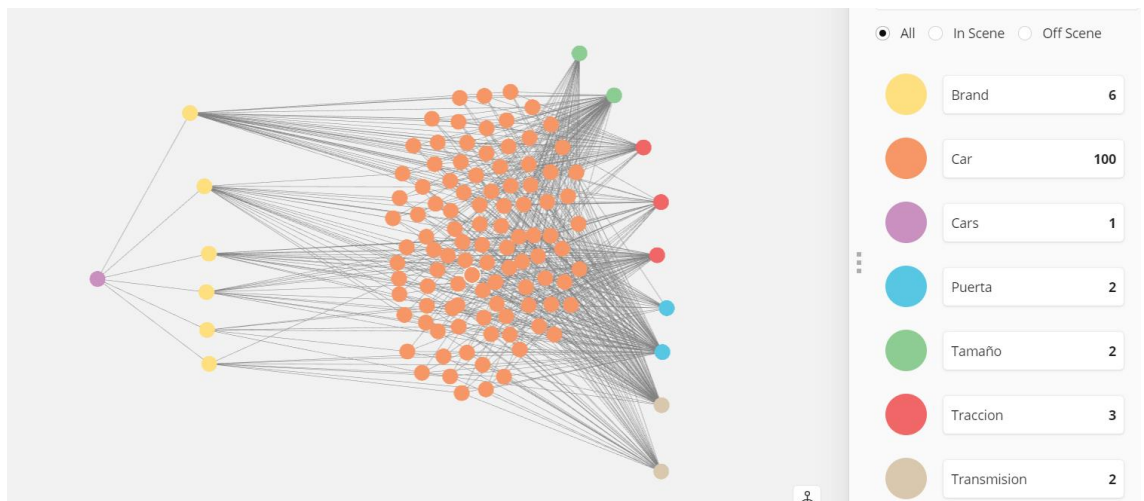


```
remove c.marca,c.puertas,c.traccion,c.transmision,c.`tamaño`
```

Algunos datos números se había guardado como texto por lo que realice la conversión:

```
MATCH (c:Car) where c.cv = toString(c.cv) set c.cv = toInteger(c.cv)
return c
MATCH (c:Car) where c.year = toString(c.year) set c.year = toInteger(
c.year) return c
MATCH (c:Car) where c.consumo_urbano = toString(c.consumo_urbano) set
c.consumo_urbano = toInteger(c.consumo_urbano) return c
```

Finalmente, el resultado de la base fue 11.871 nodos y 59.108 relaciones, a falta de añadir los nodos de cada usuario y la relacione entre los vehículos que le gusten. Esta es una pequeña muestra ya que no se puede representar entera:



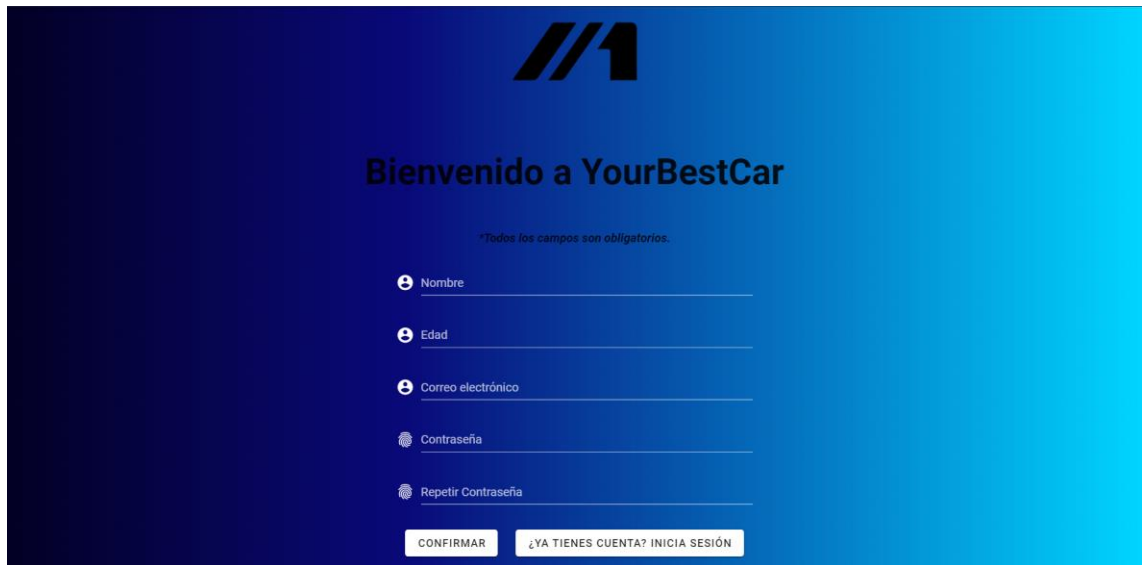
3.2 Frontend

La interfaz es la parte de la aplicación a la que menos tiempo le he dedicado ya que tenía conocimientos previos y la complejidad se encuentra en el Backend y la base.

Para esta interfaz he decidido utilizar un diseño simple e intuitivo, ya que está enfocada a cualquier grupo de edad y diseñar una interfaz compleja podría perjudicar la experiencia de usuario. Esta interfaz gira entorno a tonos azules, pero siempre predominando el blanco y el negro a la hora de presentar información en forma de texto. Creo que toda la información es visible incluso para personas con algún problema de visión.

La aplicación consta de 3 vistas principales: Login, Registro y Home.

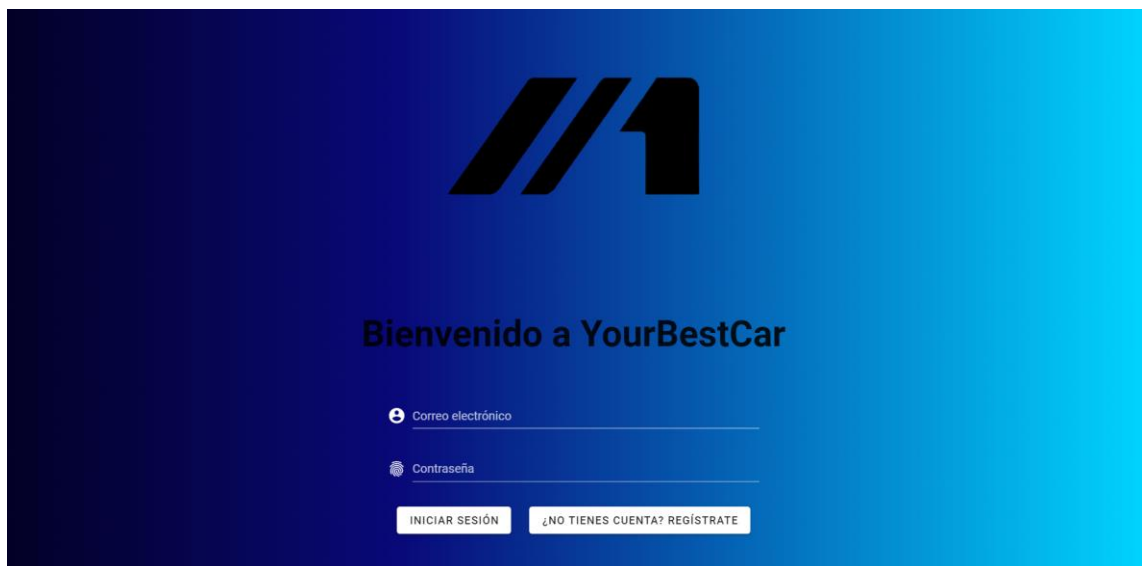
- Registro:



The screenshot shows the registration page for 'YourBestCar'. At the top is a logo consisting of two slanted parallel lines followed by the number '1'. Below the logo is the heading 'Bienvenido a YourBestCar'. A small note states '*Todos los campos son obligatorios.' Below this are five input fields, each with an icon on the left: 'Nombre' (person icon), 'Edad' (person icon), 'Correo electrónico' (envelope icon), 'Contraseña' (key icon), and 'Repetir Contraseña' (key icon). At the bottom are two buttons: 'CONFIRMAR' and '¿YA TIENES CUENTA? INICIA SESIÓN'.

Como podemos ver, los campos necesarios para el registro son el nombre, edad, email y por supuesto la contraseña. Todos los campos son obligatorios, existe comprobación para que el email tenga un formato correcto, que la edad sea mayor que 0, y que ambos campos de contraseña sean iguales. Además, existe un botón para confirmar el registro y otro que nos enviaría a la vista de Login.

- Login:



The screenshot shows the login page for 'YourBestCar'. It features the same logo as the registration page. Below the logo is the heading 'Bienvenido a YourBestCar'. There are two input fields: 'Correo electrónico' (envelope icon) and 'Contraseña' (key icon). At the bottom are two buttons: 'INICIAR SESIÓN' and '¿NO TIENES CUENTA? REGÍSTRATE'.

Vemos que la interfaz sigue la misma línea que el registro, solo que tenemos 2 campos, email y contraseña, y 2 botones, de inicio de

sesión y para cambiar la vista a Registro. En caso de contraseña o email erróneos se mostrará una advertencia.

- Home:

YourBestCar

Seleccione al menos 5 elementos de la siguiente tabla [OBTENER RECOMENDACIÓN](#)

Vehículos que te pueden interesar... Introduzca el modelo...

Modelo	Marca	Año	CV	Puertas	Tracción	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions
Tiburon	Hyundai	2006	138	2	front wheel drive	AUTOMATIC	Compact	17095	27	19	♥
New Beetle	Volkswagen	2008	150	2	front wheel drive	AUTOMATIC	Compact	20990	29	20	♥
Elise	Lotus	2009	189	2	rear wheel drive	MANUAL	Compact	43995	27	21	♥
Alero	Oldsmobile	2002	170	2	front wheel drive	AUTOMATIC	Midsize	22395	27	18	♥
Elantra Coupe	Hyundai	2013	145	2	front wheel drive	AUTOMATIC	Compact	18595	37	27	♥
Beetle	Volkswagen	2015	150	2	front wheel drive	AUTOMATIC	Compact	25895	41	31	♥
Ranger	Ford	2009	143	2	rear wheel drive	MANUAL	Compact	17990	26	21	♥
Grand Am	Pontiac	2003	175	2	front wheel drive	AUTOMATIC	Midsize	22940	27	18	♥
Camry Solara	Toyota	2006	157	2	front wheel drive	AUTOMATIC	Midsize	21855	30	20	♥
Tracker	Chevrolet	2002	127	2	all wheel drive	MANUAL	Compact	19060	23	20	♥

Rows per page: 10 1-10 of 36

Todos Los Vehículos Introduzca el modelo...

Modelo	Marca	Año	CV	Puertas	Tracción	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions
M	BMW	2001	315	2	rear wheel drive	MANUAL	Compact	44990	23	15	♥
X4	BMW	2015	240	4	all wheel drive	AUTOMATIC	Midsize	44700	28	20	♥
Alpina	BMW	2003	375	2	rear wheel drive	AUTOMATIC	Compact	136000	19	13	♥

Esta es la vista que obtenemos nada más iniciamos sesión con un usuario, en ella se muestran 2 tablas, la superior muestra un listado de los vehículos recomendados para su edad, para facilitar la experiencia de usuario y no tener que buscar entre un listado de miles de vehículos. Debajo aparece la tabla con todos los vehículos de base y todas sus propiedades. Ambas tablas permiten ordenar ascendente o descendentemente todas las propiedades, así como buscar por el nombre del modelo. Podemos ver como en la parte superior derecha tenemos un botón para cerrar sesión que nos llevaría a la vista de Login.

Además, podemos ver una alerta en color rojo que nos indica que tenemos que seleccionar al menos 5 vehículos para poder realizar una recomendación. Una vez seleccionados 5 vehículos, esta alerta cambiara de color y se habilitara el botón.


```

app.post("/registro", function (req, res) {
  var nombre = req.body.nombre;
  var edad = parseInt(req.body.edad);
  var email = req.body.email;
  var pass = req.body.pass;
  session.run('MATCH (user:Usuario {email:"' + email + '"}) RETURN user')
  .then(results => {
    if (results.records.length != 0) {
      res.json({ error: "El usuario ya estaba añadido" })
    }
    else {
      session.run('CREATE (user:Usuario {name: "' + nombre + '", email: "' + email + '", pass: "' + pass + '", edad: ' + edad + ' })')
      .then(results => {
        res.json({ correcto: "Usuario añadido correctamente" })
      })
    }
  })
})

```

Podemos ver que recibimos como parámetros los campos rellenos previamente en la vista de registro. Lo primero que se realiza es una consulta para buscar ese email en la base de datos, si esta consulta devuelve un usuario es que ya hay un usuario en la base con ese email, por lo que no puede continuarse con el registro y salta una alerta en la vista. Si por el contrario no está en uso ese email, se realiza una consulta de tipo CREATE en la que se crea un usuario estableciendo como propiedades los parámetros recibidos. Otro ejemplo, sería el login, se comprobaría esta vez que el email si está en uso y que hay un usuario con ese email y esa contraseña.

Todas las consultas tienen 2 caminos:

- **.then:** si la consulta se ha realizado, aunque hay que realizar la comprobación de que se devuelve algún dato.
- **.catch:** si la consulta no se ha podido realizar, se lanza un error, puede ser por fallo de comunicación, sintaxis incorrecta...

Una vez completada la consulta se envía la respuesta al frontend en la variable "res" y el backend continúa escuchando a próximas peticiones.

4. Algoritmo de recomendación

En este apartado vamos a explicar el funcionamiento de los sistemas de recomendación.

4.1 Algoritmo de recomendación por edad

Este es el primer algoritmo implementado se basa en realizar un filtrado inicial para que el usuario emplee menos tiempo en buscar los vehículos que le pueden gustar en base a su edad, ya que no le gusta el mismo tipo de coche a gente joven de unos 20 años, que un señor de unos 65. Por ello, la primera recomendación se establece un filtrado por etapas de edad, variando las características de las consultas a la base de datos. Después de consultar varias páginas y varios estudios acerca de que coche prefiere cada rango de edad, establecí las siguientes categorías de edad:

- Grupo 1: En este grupo se encuentran los menores de 28 años, por lo general son personas con pocos años de experiencia conduciendo, por ello no es aconsejable gran número de caballos. Principalmente buscaran su primer coche y prefieren que sea pequeño, ya que lo usaran sobre todo para trayectos cortos por ciudad. Por ello se mostrarán vehículos con las siguientes restricciones: tendrán menos de 200cv, preferiblemente de 2 puertas, posteriores al año 2000, y de tamaño Compacto o Medio.
- Grupo 2: Personas con edades comprendidas entre 29 y 37 años. Este grupo ya tiene más experiencia al volante, frecuentemente residen lejos de sus padres y seres queridos, por lo que aumentaría la cantidad de desplazamientos largos con respecto al grupo anterior. Además, muchos comienzan a tener hijos y necesitan un coche más espacioso y como para viajar. Para este grupo se mostrarán vehículos de entre 150 y 300 cv, preferiblemente que no sean de tracción delantera y que no sean compactos.
- Grupo 3: este grupo comprende a personas entre los 38 y 60 años. Este es el grupo más amplio, comprende principalmente a persona con varios hijos, que buscan vehículos amplios y cómodos para viajar en familia. Por ello es similar al grupo anterior, aunque esta vez se busca que los vehículos sean de gran tamaño.
- Grupo 4: en este están todas las personas mayores de 60 años. Este grupo es el más complicado de clasificar, ya que la mayoría no tiene cargas familiares, pero algunos quieren el vehículo para viajar, y otros simplemente para trayectos cortos. Por ellos y en base a varias encuestas, prefieren coches de pocos caballos y

preferiblemente automáticos, asique se les mostrar vehículos de menos de 150cv y automáticos.

Esto traducido en el backend seria, primero una consulta para saber la edad de la persona que tiene la sesión iniciada, una vez recibida la respuesta se lanza una query a la base estableciendo las restricciones correspondientes a cada grupo. Por ejemplo, para el segundo grupo la consulta seria la siguiente:

```
else if (edad > 28 && edad <= 37) {
  session.run('MATCH (c:Car)-[:(t:Traccion),(c:Car)-[:(y:Tamaño),(c:Car)-[:(b:Brand),(c:Car)-[:(u:Transmision),(c:Car)-[:(p:Puerta) ]where
  .then(async results => {

    results.records.forEach(function (record) {

      record._fields[0].properties["traccion"] = record._fields[1]
      record._fields[0].properties["tamano"] = record._fields[2]
      record._fields[0].properties["marca"] = record._fields[3]
      record._fields[0].properties["transmision"] = record._fields[4]
      record._fields[0].properties["puertas"] = record._fields[5]
      respuesta.cars.push(record._fields[0].properties)

    })

    res.json(respuesta)
  })
}
```

Realizamos la consulta en el “sesión.run” y guardamos los valores recibidos en un array llamado respuesta, que almacena los vehículos junto a sus propiedades.

4.2 Algoritmo de recomendación colaborativo.

Este es el algoritmo de recomendación propiamente dicho, en este caso he elegido un algoritmo de recomendación colaborativo. El filtrado colaborativo es una de las diferentes técnicas que se pueden utilizar para implementar un sistema de recomendación, como su nombre indica se basa en la recomendación implicando la colaboración entre múltiples usuarios. De manera resumida este sistema se basa en recomendar en base a usuarios que tengas gustos próximos a ti, la ventaja es que el grado de proximidad podemos variarlo como desarrollador, para que el sistema de una recomendación más exacta pero reducida una recomendación más amplia. De manera técnica seria que, si una persona A le gusta un cierto número de vehículos, que también le gustan a una persona B, es muy posible que a A le gusten los coches que le gustan a B. Para ellos como es obvio, cuantos más usuarios utilicen la aplicación, más datos tenemos y más exacto será el modelo creado para cada usuario.

En el libro mostrado en la bibliografía “Graph Powered Machine Learning” se explica cómo se debería implementar un sistema de recomendación de este tipo. Como claves que deberíamos tener en cuenta a la hora de implementarlo, una seria el inicio en frio, esto es que no hay suficientes datos de usuarios por lo que es posible que no se encuentre ningún usuario similar a ti y por lo tanto la recomendación no

se pueda llevar a cabo, este problema tendría una solución de crear unos usuarios inicial, que abarcasen la mayoría de vehículos para así reducir las posibilidades de una recomendación vacía o nula, además se ha añadido una obligación de “dar like” al menos a 5 vehículos para obtener una recomendación y así tener unos datos mínimos sobre los gustos de esa persona.

Para implementar este sistema, se debería crear una matriz vectorizando los gustos de cada usuario, estos vectores pueden tener valores binarios o por ejemplo de 0 a 10 para expresar el grado en que nos gusta algo, se realiza una comparación de vectores con técnicas como la comparación de cosenos, y se obtiene un listado de los más próximos, con este método podemos elegir la cantidad de vecinos que queremos, si tuviésemos problemas de arranque en frío sería tan sencillo como elegir un mayor número de vecinos, esto nos daría una recomendación correcta pero no tan exacta.

Para calcular la matriz en el caso de tener un gran número de usuarios, necesitaríamos una estructura de datos potentes y un lenguaje que nos permitiese almacenar esa información, por desgracia el lenguaje utilizado es JavaScript, que carece de esta potencia, por lo que he elegido una variante para crear el algoritmo de recomendación.

Partimos de una consulta para obtener los vehículos que le gustan a cualquier usuario que tiene algún vehículo con común a nosotros, es decir al usuario A le gustan los vehículos 1,2,3 y al usuario B le gustan 3,4,5, esta consulta para el usuario a nos devolvería los vehículos 4 y 5. Guardamos la respuesta de la base en un array de vehículos llamado respuesta:

```
session.run('MATCH (ser:Usuario{email:"" + email + ""})-[:LIKES]-(a:Car)-[:LIKES]-(pep:Usuario) Match (pep)-[:LIKES]-(c:Car) where not exists((se
.then(results => {
  results.records.forEach(function (record) {
    record._fields[0].properties["traccion"] = record._fields[1]
    record._fields[0].properties["tamano"] = record._fields[2]
    record._fields[0].properties["marca"] = record._fields[3]
    record._fields[0].properties["transmision"] = record._fields[4]
    record._fields[0].properties["puertas"] = record._fields[5]
    respuesta.cars.push(record._fields[0].properties)
  })
})
```

Ahora procederemos a optimizar esta respuesta en base a los gustos del usuario, para ellos obtendremos un listado de los vehículos que le gustan al usuario, a partir de ahí el primer paso es comprobar las puertas.

Comprobamos de los vehículos que le gustan cuantos son de 2 puertas y cuantos son de 4 puertas. Si alguno supera el 65% se eliminarán todos los vehículos con la propiedad contraria, es decir si por ejemplo un 72% de los vehículos que me gustan son de 2 puertas, se eliminarían de la consulta realizada previamente todos los de 4 puertas.

Para la siguiente etapa realizaremos el mismo proceso, pero esta vez con la propiedad “Transmision” que puede tomar los valores

“AUTOMATIC” o “MANUAL”, si alguno de los dos aparece más del 65% de las veces se eliminara el contrario, como mencionamos en la etapa anterior.

A continuación, se repite el proceso con la propiedad “Tamaño” que en este caso puede tomar 3 valores: “compact”, “midsize” o “large”. Al igual que anteriormente, si alguno de los valores aparece mayoritariamente se eliminarán los otros dos. En este caso el porcentaje elegido es del 55% lo que significaría que más de la mitad son de un tipo.

Al igual que para la etapa anterior, en esta se trata la propiedad “Tracción” que puede tomar 3 valores: “front Wheel drive”, “rear Wheel drive” o “all Wheel drive”. Como ya hemos mencionado, si alguno de los valores aparece más del 55% de las veces se eliminarán el resto.

Una vez finalizados estos procesos se envía la respuesta hacia en frontend para representarla en forma de tabla.

5. Análisis de resultados

En este apartado voy a realizar una exposición de distintos ejemplos para distintos grupos de edad con distintos grupos, para ello ha utilizado la aplicación una persona de cada grupo de edad.

5.1 Ejemplo 1

Este primer ejemplo es una persona perteneciente al primer grupo de edad, de 22 años, una vez registrado comienza a valorar los vehículos que le gustan. Vemos como sería la tabla de recomendación en función de la edad para el primer grupo:

Vehículos que te pueden interesar...							Introduzca el modelo...				
Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions
Tiburon	Hyundai	2006	138	2	front wheel drive	AUTOMATIC	Compact	17095	27	19	♥
New Beetle	Volkswagen	2008	150	2	front wheel drive	AUTOMATIC	Compact	20990	29	20	♥
Elise	Lotus	2009	189	2	rear wheel drive	MANUAL	Compact	43995	27	21	♥
Alero	Oldsmobile	2002	170	2	front wheel drive	AUTOMATIC	Midsize	22395	27	18	♥
Elantra Coupe	Hyundai	2013	145	2	front wheel drive	AUTOMATIC	Compact	18595	37	27	♥
Beetle	Volkswagen	2015	150	2	front wheel drive	AUTOMATIC	Compact	25895	41	31	♥
Ranger	Ford	2009	143	2	rear wheel drive	MANUAL	Compact	17990	26	21	♥
Grand Am	Pontiac	2003	175	2	front wheel drive	AUTOMATIC	Midsize	22940	27	18	♥
Camry Solara	Toyota	2006	157	2	front wheel drive	AUTOMATIC	Midsize	21855	30	20	♥
Tracker	Chevrolet	2002	127	2	all wheel drive	MANUAL	Compact	19060	23	20	♥

Rows per page: 10 1-10 of 36

Se muestran 36 vehículos que cumplen las características para este grupo de edad.

"c.model"	"t.name"	"y.name"	"b.name"	"u.name"	"p.numero"
"Paseo"	"front wheel drive"	"Compact"	"Toyota"	"MANUAL"	2
"Juke"	"front wheel drive"	"Compact"	"Nissan"	"AUTOMATIC"	4
"3"	"front wheel drive"	"Compact"	"Mazda"	"MANUAL"	4
"Rio"	"front wheel drive"	"Compact"	"Kia"	"AUTOMATIC"	4
"Escort"	"front wheel drive"	"Compact"	"Ford"	"MANUAL"	2
"1 Series"	"rear wheel drive"	"Compact"	"BMW"	"MANUAL"	2
"500"	"front wheel drive"	"Compact"	"FIAT"	"MANUAL"	2
"Celica"	"front wheel drive"	"Compact"	"Toyota"	"AUTOMATIC"	2
"CR-Z"	"front wheel drive"	"Compact"	"Honda"	"AUTOMATIC"	2
"Swift"	"front wheel drive"	"Compact"	"Suzuki"	"MANUAL"	2
"C30"	"front wheel drive"	"Compact"	"Volvo"	"MANUAL"	2
"Focus"	"front wheel drive"	"Compact"	"Ford"	"MANUAL"	4
"MX-5 Miata"	"rear wheel drive"	"Compact"	"Mazda"	"AUTOMATIC"	2
"Mazdaspeed MX-5 Miata"	"rear wheel drive"	"Compact"	"Mazda"	"MANUAL"	2
"Elise"	"rear wheel drive"	"Compact"	"Lotus"	"MANUAL"	2

En la imagen superior se muestra el listado de los vehículos que le han gustado al usuario y sus características, vemos que la mayoría de vehículos son “front Wheel drive”, “MANUAL” y “2 puertas”, además todos son “Compact”.

Por lo que si pedimos la recomendación el resultado será el siguiente:

Tu recomendacion...						Introduzca el modelo...						
Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions	
Civic CRX	Honda	1990	62	2	front wheel drive	MANUAL	Compact	2000	44	36	♥	
G5	Pontiac	2007	148	2	front wheel drive	MANUAL	Compact	14775	31	22	♥	
RSX	Acura	2004	160	2	front wheel drive	MANUAL	Compact	21100	30	24	♥	
Sunfire	Pontiac	2003	140	2	front wheel drive	MANUAL	Compact	14930	30	22	♥	
Cobalt	Chevrolet	2008	148	2	front wheel drive	MANUAL	Compact	14410	33	24	♥	
tC	Scion	2014	179	2	front wheel drive	MANUAL	Compact	21400	31	23	♥	
Rows per page:									All	1-6 of 6	<	>

Podemos ver cómo nos recomienda 6 vehículos, los cuales cumplen claramente los gustos del usuario. A medida que vaya añadiendo más vehículos a sus gustos, o más usuarios utilicen la aplicación, mas vehículos recibirá en la recomendación.

5.2 Ejemplo 2

Para este ejemplo hemos pedido a una persona de 31 años que utilice la aplicación. Una vez registrado a comenzado a añadir vehículos a sus gustos. Vemos como la primera tabla de recomendación en función de la edad varia con respecto a la del ejemplo anterior:

Vehiculos que te pueden interesar...

Introduzca el modelo...












Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions
Forester	Subaru	2015	250	4	all wheel drive	AUTOMATIC	Midsize	33095	28	23	
RAV4 Hybrid	Toyota	2016	194	4	all wheel drive	AUTOMATIC	Midsize	28370	31	34	
Passport	Honda	2000	205	4	rear wheel drive	MANUAL	Midsize	2452	18	15	
Sportvan	Chevrolet	1994	200	2	rear wheel drive	AUTOMATIC	Midsize	2000	16	11	
Montero Sport	Mitsubishi	2002	197	4	rear wheel drive	AUTOMATIC	Midsize	27607	20	15	
EX35	Infiniti	2008	297	4	rear wheel drive	AUTOMATIC	Midsize	31900	24	17	
Discovery Sport	Land Rover	2015	240	4	all wheel drive	AUTOMATIC	Midsize	37070	26	20	
Bravada	Oldsmobile	2002	270	4	all wheel drive	AUTOMATIC	Midsize	34745	19	14	
Sorento	Kia	2015	290	4	all wheel drive	AUTOMATIC	Midsize	38500	24	18	
Rainier	Buick	2005	275	4	all wheel drive	AUTOMATIC	Midsize	35610	19	14	
<div>Rows per page: 10 1-10 of 164 < ></div>											

En este caso se muestran 164 vehículos, recomendados para ese grupo de edad.

Si mostramos un resultado de los vehículos que le gustan a ese usuario, obtenemos lo siguiente:

"c.model"	"t.name"	"y.name"	"b.name"	"u.name"	"p.numero"
"3"	"front wheel drive"	"Compact"	"Mazda"	"MANUAL"	4
"NV200"	"front wheel drive"	"Compact"	"Nissan"	"AUTOMATIC"	4
"S40"	"all wheel drive"	"Compact"	"Volvo"	"AUTOMATIC"	4
"350Z"	"rear wheel drive"	"Compact"	"Nissan"	"AUTOMATIC"	2
"370Z"	"rear wheel drive"	"Compact"	"Nissan"	"AUTOMATIC"	2
"300ZX"	"rear wheel drive"	"Compact"	"Nissan"	"MANUAL"	2
"SLK-Class"	"rear wheel drive"	"Compact"	"Mercedes-Benz"	"AUTOMATIC"	2
"B-Class Electric Drive"	"front wheel drive"	"Compact"	"Mercedes-Benz"	"AUTOMATIC"	4
"C43 AMG"	"rear wheel drive"	"Compact"	"Mercedes-Benz"	"AUTOMATIC"	4
"Veracruz"	"all wheel drive"	"Midsize"	"Hyundai"	"AUTOMATIC"	4
"M30"	"rear wheel drive"	"Midsize"	"Infiniti"	"AUTOMATIC"	2
"Bronco"	"all wheel drive"	"Midsize"	"Ford"	"MANUAL"	2
"Durango"	"rear wheel drive"	"Large"	"Dodge"	"AUTOMATIC"	4
"Venture"	"all wheel drive"	"Midsize"	"Chevrolet"	"AUTOMATIC"	4
"8 Series"	"rear wheel drive"	"Midsize"	"BMW"	"AUTOMATIC"	2
"X4"	"all wheel drive"	"Midsize"	"BMW"	"AUTOMATIC"	4
"A4 allroad"	"all wheel drive"	"Midsize"	"Audi"	"AUTOMATIC"	4
"A6"	"all wheel drive"	"Midsize"	"Audi"	"AUTOMATIC"	4
"S3"	"all wheel drive"	"Compact"	"Audi"	"AUTOMATIC"	4
"A3"	"front wheel drive"	"Compact"	"Audi"	"AUTOMATIC"	4
"1 Series"	"rear wheel drive"	"Compact"	"BMW"	"MANUAL"	2

Si obtenemos la recomendación para este usuario, podemos ver que el único gusto claro y mayoritario es que prefiere los vehículos "Compact" sobre el resto, por lo que la recomendación seguirá esa tendencia. En la siguiente imagen se muestran 10 de los 41 vehículos que recomienda, aunque todos siguen la misma línea de recomendación:

Tu recomendacion...						Introduca el modelo...						
Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions	
Paseo	Toyota	1995	100	2	front wheel drive	MANUAL	Compact	2000	33	25		
Juke	Nissan	2015	188	4	front wheel drive	AUTOMATIC	Compact	25240	32	28		
Rio	Kia	2015	138	4	front wheel drive	AUTOMATIC	Compact	15190	37	27		
Escort	Ford	2001	130	2	front wheel drive	MANUAL	Compact	12340	31	23		
500	FIAT	2015	160	2	front wheel drive	MANUAL	Compact	26595	34	28		
Celica	Toyota	2003	140	2	front wheel drive	AUTOMATIC	Compact	18140	33	25		
CR-Z	Honda	2014	130	2	front wheel drive	AUTOMATIC	Compact	20645	39	36		
Swift	Suzuki	1999	79	2	front wheel drive	MANUAL	Compact	2000	39	33		
C30	Volvo	2011	227	2	front wheel drive	MANUAL	Compact	24700	29	21		
Focus	Ford	2015	160	4	front wheel drive	MANUAL	Compact	18460	36	26		
Rows per page: 10 1-10 of 41 < >												

5.3 Ejemplo 3

Para este ejemplo he elegido una persona de 55 años, perteneciente al tercer grupo de edad, por lo tanto, tu tabla de recomendación de edad, será distinta a las anteriores:

Vehiculos que te pueden interesar...















Introduca el modelo...

Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions		
Ridgeline	Honda	2013	250	4	all wheel drive	AUTOMATIC	Large	37380	21	15			
Tahoe Limited/Z71	Chevrolet	2000	255	4	all wheel drive	AUTOMATIC	Large	3465	15	11			
Ninety-Eight	Oldsmobile	1994	170	4	front wheel drive	AUTOMATIC	Large	2000	25	17			
Pathfinder	Nissan	2015	260	4	front wheel drive	AUTOMATIC	Large	29510	27	20			
Caprice	Chevrolet	1994	260	4	rear wheel drive	AUTOMATIC	Large	2000	23	15			
Sierra Classic 1500	GMC	1999	200	2	rear wheel drive	AUTOMATIC	Large	4439	17	13			
Envoy XUV	GMC	2005	275	4	all wheel drive	AUTOMATIC	Large	36115	17	12			
420-Class	Mercedes-Benz	1990	201	4	rear wheel drive	AUTOMATIC	Large	2000	17	14			
Park Avenue	Buick	2003	240	4	front wheel drive	AUTOMATIC	Large	39725	25	16			
300-Class	Mercedes-Benz	1991	177	4	rear wheel drive	AUTOMATIC	Large	2232	19	15			
Rows per page:									10		1-10 of 83		

En este caso se muestran 83 vehículos, el usuario comienza a valorar vehículos y el resultado sería este:

"c.model"	"t.name"	"y.name"	"b.name"	"u.name"	"p.numero"
"LSS"	"front wheel drive"	"Large"	"Oldsmobile"	"AUTOMATIC"	4
"Aspen"	"rear wheel drive"	"Large"	"Chrysler"	"AUTOMATIC"	4
"Sienna"	"all wheel drive"	"Large"	"Toyota"	"AUTOMATIC"	4
"Grand Voyager"	"front wheel drive"	"Large"	"Chrysler"	"AUTOMATIC"	4
"Astro"	"all wheel drive"	"Large"	"Chevrolet"	"AUTOMATIC"	2
"Q7"	"all wheel drive"	"Large"	"Audi"	"AUTOMATIC"	4
"R-Class"	"all wheel drive"	"Large"	"Mercedes-Benz"	"AUTOMATIC"	4
"LS 430"	"rear wheel drive"	"Large"	"Lexus"	"AUTOMATIC"	4
"M35"	"rear wheel drive"	"Large"	"Infiniti"	"AUTOMATIC"	4
"Five Hundred"	"front wheel drive"	"Large"	"Ford"	"AUTOMATIC"	4
"560-Class"	"rear wheel drive"	"Large"	"Mercedes-Benz"	"AUTOMATIC"	2
"Impala"	"front wheel drive"	"Large"	"Chevrolet"	"AUTOMATIC"	4
"Q60 Coupe"	"rear wheel drive"	"Midsize"	"Infiniti"	"MANUAL"	2
"LX 450"	"all wheel drive"	"Midsize"	"Lexus"	"AUTOMATIC"	4
"Optima"	"front wheel drive"	"Midsize"	"Kia"	"AUTOMATIC"	4
"Amanti"	"front wheel drive"	"Midsize"	"Kia"	"AUTOMATIC"	4
"Sorento"	"all wheel drive"	"Midsize"	"Kia"	"AUTOMATIC"	4
"Cayman S"	"rear wheel drive"	"Compact"	"Porsche"	"MANUAL"	2
"Prius Prime"	"front wheel drive"	"Compact"	"Toyota"	"AUTOMATIC"	4
"B-Series"	"all wheel drive"	"Compact"	"Mazda"	"MANUAL"	2
"S40"	"all wheel drive"	"Compact"	"Volvo"	"AUTOMATIC"	4

Como podemos ver la mayoría de vehículos son "4 puertas", al igual que de tamaño "Large" y Transmisión "AUTOMATIC", por lo que, si obtenemos la recomendación, obtenemos 51 vehículos que siguen estas características:

Tu recomendacion...						Introduca el modelo...							
Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions		
Panamera	Porsche	2015	570	4	all wheel drive	AUTOMATIC	Large	200500	24	15			
Flying Spur	Bentley	2014	616	4	all wheel drive	AUTOMATIC	Large	200500	20	12			
Durango	Dodge	2015	290	4	rear wheel drive	AUTOMATIC	Large	36895	25	18			
Aurora	Oldsmobile	2001	250	4	front wheel drive	AUTOMATIC	Large	34794	23	15			
Sixty Special	Cadillac	1993	200	4	front wheel drive	AUTOMATIC	Large	2000	23	15			
A8	Audi	2016	240	4	all wheel drive	AUTOMATIC	Large	85200	36	24			
300M	Chrysler	2002	250	4	front wheel drive	AUTOMATIC	Large	32065	24	16			
Intrepid	Dodge	2002	244	4	front wheel drive	AUTOMATIC	Large	27055	24	16			
Crown Victoria	Ford	2009	224	4	rear wheel drive	AUTOMATIC	Large	29205	24	16			
LeSabre	Buick	2003	205	4	front wheel drive	AUTOMATIC	Large	25645	27	17			
Rows per page:									10		1-10 of 51		

5.4 Ejemplo 4

Para el último ejemplo he elegido a una persona de 68 años, como es normal la tabla que muestra la recomendación en función de la edad es distinta a las 3 anteriores ya que pertenece al último grupo de edad:

Vehiculos que te pueden interesar...							Introduca el modelo...					
Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions	
Previa	Toyota	1995	138	2	rear wheel drive	AUTOMATIC	Compact	2000	20	15		
Accent	Hyundai	2014	138	4	front wheel drive	AUTOMATIC	Compact	17395	37	27		
Prius c	Toyota	2014	99	4	front wheel drive	AUTOMATIC	Compact	20030	46	53		
Prius v	Toyota	2015	134	4	front wheel drive	AUTOMATIC	Midsize	26675	40	44		
Celebrity	Chevrolet	1990	135	4	front wheel drive	AUTOMATIC	Midsize	2000	27	17		
CX-3	Mazda	2016	146	4	front wheel drive	AUTOMATIC	Compact	19960	35	29		
Elantra Coupe	Hyundai	2013	145	2	front wheel drive	AUTOMATIC	Compact	18595	37	27		
NV200	Nissan	2015	131	4	front wheel drive	AUTOMATIC	Compact	21710	26	24		
Matrix	Toyota	2011	132	4	front wheel drive	AUTOMATIC	Compact	19685	32	25		
MR2 Spyder	Toyota	2003	138	2	rear wheel drive	AUTOMATIC	Compact	25645	30	22		
Rows per page:									10	1-10 of 55	<	>

Comienza a añadir vehículos a sus gustos y obtenemos el listado:

"c.model"	"t.name"	"y.name"	"b.name"	"u.name"	"p.numero"
"200SX"	"front wheel drive"	"Compact"	"Nissan"	"MANUAL"	2
"740"	"rear wheel drive"	"Midsize"	"Volvo"	"MANUAL"	4
"B-Series Pickup"	"rear wheel drive"	"Compact"	"Mazda"	"MANUAL"	2
"Cube"	"front wheel drive"	"Compact"	"Nissan"	"MANUAL"	4
"240SX"	"rear wheel drive"	"Compact"	"Nissan"	"MANUAL"	2
"500"	"front wheel drive"	"Compact"	"FIAT"	"MANUAL"	2
"500L"	"front wheel drive"	"Compact"	"FIAT"	"MANUAL"	4
"M3"	"rear wheel drive"	"Midsize"	"BMW"	"MANUAL"	4
"Z4"	"rear wheel drive"	"Compact"	"BMW"	"MANUAL"	2
"X3"	"all wheel drive"	"Midsize"	"BMW"	"AUTOMATIC"	4
"Versa Note"	"front wheel drive"	"Compact"	"Nissan"	"AUTOMATIC"	4
"Custom Cruiser"	"rear wheel drive"	"Large"	"Oldsmobile"	"AUTOMATIC"	4
"Soul"	"front wheel drive"	"Compact"	"Kia"	"AUTOMATIC"	4
"MR2 Spyder"	"rear wheel drive"	"Compact"	"Toyota"	"AUTOMATIC"	2
"CX-3"	"front wheel drive"	"Compact"	"Mazda"	"AUTOMATIC"	4
"Prius v"	"front wheel drive"	"Midsize"	"Toyota"	"AUTOMATIC"	4
"Accent"	"front wheel drive"	"Compact"	"Hyundai"	"AUTOMATIC"	4

Como podemos ver la única restricción clara es que le gustan los vehículos de tamaño "Compact" y que le es indiferente si es automático o manual y si tiene 2 o 4 puertas. Por lo que la recomendación nos muestra 79 vehículos, todos de tipo "Compact":

Tu recomendacion...						Introduce el modelo...					
Modelo	Marca	Año	CV	Puertas	Traccion	Cambio	Tamaño	Precio	Consumo ExtraUrbano	Consumo Urbano	Actions
New Beetle	Volkswagen	2008	150	2	front wheel drive	AUTOMATIC	Compact	20990	29	20	♥
Tiburon	Hyundai	2006	138	2	front wheel drive	AUTOMATIC	Compact	17095	27	19	♥
Z3	BMW	2000	193	2	rear wheel drive	MANUAL	Compact	4976	24	17	♥
V12 Vantage S	Aston Martin	2015	565	2	rear wheel drive	AUTOMATIC	Compact	182395	18	12	♥
Viper	Dodge	2015	645	2	rear wheel drive	MANUAL	Compact	107995	21	12	♥
Vitara	Suzuki	2002	127	2	rear wheel drive	AUTOMATIC	Compact	16599	23	20	♥
Blazer	Chevrolet	2003	190	2	rear wheel drive	MANUAL	Compact	20745	20	14	♥
124 Spider	FIAT	2017	160	2	rear wheel drive	MANUAL	Compact	27495	35	26	♥
G5	Pontiac	2007	148	2	front wheel drive	MANUAL	Compact	14775	31	22	♥
Forte	Kia	2015	173	2	front wheel drive	AUTOMATIC	Compact	19590	34	25	♥

Rows per page: 10 1-10 of 79 < >

Esto sería el conjunto de ejemplo para cada grupo de edad establecido en la aplicación, estos grupos como podemos ver no siempre cuadran con los gustos de la persona, pero sirve como aproximación, ya que la recomendación siempre coincide con el conjunto de gustos de cada usuario independientemente de su edad.

6.DAFO

En este apartado se realiza un análisis DAFO que nos permite analizar sus características internas y su situación externa:

- **Debilidades:** El algoritmo de búsqueda podría ser mucho más complejo y preciso, ya que con la información del libro “Graph Powered Machine Learning” hay muchos tipos de algoritmo y de hibridación para completar las carencias de alguno. Y tiene que ver con esto el lenguaje usado, ya que JavaScript tiene más restricciones de memoria y hay mucha menos documentación.
- **Amenazas:** En todo el desarrollo de la aplicación estuve buscando información sobre alguna aplicación similar para obtener ideas y no encontré nada. Por lo que creo que no hay ningún sistema de recomendación de vehículos, simplemente hay sistemas de búsqueda que en base a unos parámetros (km, color, cv, año...) te devuelven un listado de vehículos, pero nada en base a tus propios gustos o el de otros usuarios similares a ti.
- **Fortaleza:** La gran fortaleza de esta aplicación es que con lo sencilla que es como ahorra el tiempo y facilita la vida al usuario que la use, ya que comprar un vehículo siempre es un quebradero de cabeza. Otra sería la base ya que de la manera en la que está estructurada sería muy sencillo aportar más información leyendo otros ficheros, incluso se podrían añadir imágenes.
- **Oportunidades:** el mercado de los coches, está cambiando y los países quiere electrificar su parque móvil, por lo que todo el mundo está pensando en cambiar de vehículo y cuál será el mejor para los próximos años.

Cuando comencé a desarrollar este proyecto pensé que me iba a llevar más tiempo documentarme y buscar información para comprender el problema, que en si lo que iba a ser el desarrollo, pero estaba equivocado, ya que la implementación del algoritmo de recomendación y el creación y estructuración de la base puede llegar a tomar mucho tiempo.

Respecto al tema de Cypher y Neo4j, estoy muy contento ya que conocido un nuevo lenguaje que se manejar con fluidez y un nuevo sistema de bases de datos que se sale de lo habitual como, MySQL, Oracle... Creo que realizar los cursos que ofrece Neo4j de introducción son una gran inversión de tiempo, ya que en pocas horas te enseña consultas básicas y te ayuda a entender la lógica que sigue el lenguaje. Además, implementa el modo sandbox que te permite realizar pruebas sin modificar tu base de datos original.

Otro punto que creo que es importante, es que una aplicación de este tipo realizada en grupo o con más tiempo, podría tener un potencian increíble, ya que hay infinitas ramas de exploración y por donde aumentar y mejorar la funcionalidad del sistema.

7. Líneas de futuro

Como ya hemos comentado en este informe, el sector de los vehículos está cambiando, y se está produciendo una transición hacia lo eléctrico, por lo que la gente está pensando en cambiar su coche actual. Durante mi búsqueda encontré páginas y reviews en YouTube que mostraban un resumen de lo que les había parecido un coche en concreto a la gente que lo había probado, pero no encontré nada similar a este proyecto.

Creo que mi aplicación puede tomar varios caminos, el más claro que veo es que sea una página/aplicación de uso libre, con una base de datos constantemente actualizada, y que un usuario en el momento que quiera pueda consultarla, sin mostrar preferencias de unas marcas sobre otras. Se pondrían añadir funcionalidades como mostrar opiniones de texto para cada vehículo compartidas por otros usuarios, o poner reviews del vehículo para ayudar al usuario a tomar una decisión.

La introducción de fotografías de cada vehículo podría ser una gran línea de expansión, pero habría que estudiarlo en profundidad, ya que la base que se está utilizando ahora no muestra todos los vehículos que existen y podemos encontrar problemas de rendimiento en algunas consultas. En el mundo existen unas 155 marcas de vehículos, cada marca tiene unas decenas de modelos, y cada modelo puede tener versiones en distintos años, si a esto le sumamos un par de imágenes por cada modelo, las consultas podrían demorarse tanto tiempo que impediría el correcto uso de la aplicación. También se debería tener una correcta infraestructura capaz de almacenar ese volumen de datos y de mantener el sistema activo en el tiempo. Por ello estos planes de crecimiento tienen que estar bien estudiados y documentados para comprobar la viabilidad de su ejecución.

Igual que he mencionado la introducción de fotografías, también se podría aumentar la cantidad de propiedades como: número de marchas, medidas, volumen del maletero y un largo etc. Al igual que pasaba con la imágenes, esta expansión también llevaría un estudio previo, ya que habría que obtener cuales son las propiedades más importantes para cada usuario a la hora de realizar la recomendación, así como la manera de mostrar los datos, ya que por ejemplo si se muestran 20 propiedades, en una tabla no se vería la información clara y podría causar dudas al usuario, además se perdería la esencia de simplicidad de la aplicación, y de esta manera habría gente reacia a usarla por su complejidad.

En resumen, creo que esta aplicación tiene un gran potencial tal y como esta simplemente optimizando el algoritmo de recomendación, ya que no he encontrado un sistema de recomendación basado en vehículos en internet, por lo que podría ser la aplicación sobre este tema. Pero por otro lado la aplicación puede mejorar mucho como ya he mencionado anteriormente, por distintos caminos, pero toda expansión que se haga tiene que tener un exhaustivo

estudio previo de viabilidad, ya que se realizan las modificaciones en bruto puede ser que la aplicación deje de funcionar y sea demasiado tarde para volver atrás.

8. Lecciones aprendidas

Esta asignatura me ha ayudado a conocerme a mí mismo y saber que soy capaz de desarrollar un proyecto de 0. La idea de la asignatura era montar un sistema de recomendación con Neo4j y a partir de ahí tenías libertad absoluta para hacer lo que quieras. Lo mejor que puede hacer fue establecerme unos plazos en el desarrollo de la aplicación ya que me ayudo a mantener un control de como llevaba la aplicación, si iba retrasado o bien de tiempo. Creo que lo fácil hubiese sido escoger un sistema de recomendación de música o películas, ya que hay mucha información en internet y no sería nada nuevo, por lo que creo que la elección del tema ha sido acertada y me ha ayudado a superarme a mí mismo.

Gracias al libro “Graph Powered Machine Learning” he obtenido un gran conocimiento del tema de sistema de recomendación, ya te explica en profundidad todo lo necesario para montar un sistema, desde mejores lenguajes para usar hasta el funcionamiento de los diferentes algoritmos que se pueden implementar. La verdad que es una lectura que recomiendo a todo el mundo antes de empezar a desarrollar un sistema de este tipo, al principio puede parecer que no entiendes nada, pero después de realizar una primera lectura y empezar a desarrollar tu sistema, vas a entender muchas cosas que sabrás hacerlas gracias a haberte leído el libro.

Otra cosa de la que estoy muy contento es de haber sido capaz de solucionar todos los problemas que me he encontrado durante el desarrollo, ya sean problemas de código, de funcionamiento... He sido capaz de solucionarlos todos ya sea por ensayo y error o buscando en distintos tipos de documentación y es algo de lo que estoy muy satisfecho.

La idea de lo que quería hacer la tenía desde el principio, pero a medida que fui implementado me encontré con obstáculos que no tenían solución por lo que opté por otros caminos que me han llevado a completar el proyecto de manera exitosa.

En general estoy muy contento con todo lo que he aprendido, ya sea acerca de sistemas de recomendación como he mencionado antes, como con explorar nuevas tecnologías, ya que Neo4j y su lenguaje Cypher eran nuevos para mí y al final he sido capaz de trabajar con ellos a la perfección y adaptarme en poco tiempo a su uso.

9. Bibliografía

1. **Neo4j:** <https://neo4j.com/blog/add-users-node-js-react-js-neo4j-movie-app/>
<https://neo4j.com/docs/cypher-manual/current/>
<https://www.diegocalvo.es/tutorial-neo4j-espanol/>
<http://blog.jortilles.com/neo4j/>
2. **Vuetify:** <https://vuetifyjs.com/en/introduction/why-vuetify/#getting-started>
3. **Nodejs:** <https://nodejs.org/docs/latest-v13.x/api/>
https://www.youtube.com/watch?v=snjnJCZhXUM&ab_channel=TraversyMedia
4. **Graph Powered Machine Learning:** <https://neo4j.com/graph-powered-machine-learning-manning-book/>
5. **Enlaces de interés:**
<https://www.sciencedirect.com/science/article/pii/S1110866515000341>
<https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada>
<https://github.com/citruz/movies4j>
<https://github.com/neo4j-examples/movies-javascript-bolt/tree/main/src/models>
https://www.youtube.com/watch?v=snjnJCZhXUM&ab_channel=TraversyMedia