UNIVERSIDAD
DE GRANADA

Gestión de Información en la Web

Desarrollo de un Sistema de Recomendación basado en filtrado colaborativo

Práctica 4

Víctor Vázquez Rodríguez
victorvazrod@correo.ugr.es
76664636R

Máster universitario en Ingeniería Informática

Curso 2019/20

Índice

1. Introducción **2**

2. Algoritmo **3**

3. Interfaz **5**

3.1. Manual de uso 6

4. Referencias **7**

1. Introducción

Para esta práctica, se ha desarrollado un sistema de recomendación de películas basado en el método del filtrado colaborativo. Las recomendaciones se obtienen a partir de un conjunto de valoraciones de películas de la colección *MovieLens* junto con las valoraciones introducidas por el usuario del sistema.

En concreto, se ha desarrollado una aplicación web con el lenguaje de programación Go y el *framework* Gin, el cuál está pensado para la creación de este tipo de aplicaciones. Además, los datos de los ficheros se cargan a una pequeña base de datos SQLite para facilitar el trabajo con los mismos desde la aplicación. La conexión a esta base de datos y el trabajo con la misma desde la aplicación se ha realizado usando GORM.

En las siguientes secciones de este documento, se expondrán el algoritmo y las fórmulas utilizadas para obtener las recomendaciones y se mostrará también la interfaz de usuario desarrollada para el sistema.

2. Algoritmo

El objetivo de un sistema de recomendación es presentar a un usuario elementos que no ha visto/consumido y que pueden ser de su interés. Una manera de conseguir esto es mediante las llamadas técnicas de filtrado colaborativo, las cuáles se basan en las acciones de los usuarios (dar "me gusta/"no me gusta", puntuar del 1 al 5, ...) para obtener las recomendaciones, aplicando modelos como KNN (*K Nearest Neighbors*, que es el que usaremos en esta práctica.

La idea principal de KNN es buscar los k elementos o vecinos más similares a uno dado y, a partir de ellos, obtener las recomendaciones. Estos elementos pueden ser ítems (basado en ítems) o usuarios del sistema (basado en usuarios). Nosotros aplicaremos KNN basado en usuarios para nuestro sistema de recomendación, de forma que buscamos los usuarios más parecidos al usuario activo para recomendarle las películas que les han gustado a estos otros usuarios.

Las películas son valoradas del 1 al 5, por lo que tenemos que tener en cuenta, además, que distintos usuarios valoran un ítem que les ha gustado de forma diferente, por lo que, lo primero que hacemos, es calcular un valor para equilibrar esos desajustes de escala e interpretación. La fórmula para realizar este cálculo se muestra a continuación, donde S_u es el conjunto de ítems valorados por el usuario u y n_{S_u} es el número de ítems del mismo.

$$\bar{r}(u) = \frac{\sum_{i \epsilon S_u} r(u,i)}{n_{S_u}} \tag{1}$$

Una vez obtenido este valor para los usuarios, podemos calcular la similitud entre un usuario u y otro usuario v con la ecuación (2), donde $r(u,i)$ es la valoración dada por el usuario u para el ítem i.

$$sim(u,v) = \frac{\sum_{i \epsilon S_{uv}} (r(u,i) - \bar{r}(u))(r(v,i) - \bar{r}(v))}{\sqrt{\sum_{i \epsilon S_{uv}} (r(u,i) - \bar{r}(u))^2} \sqrt{\sum_{i \epsilon S_{uv}} (r(v,i) - \bar{r}(v))^2}} \tag{2}$$

Usando esta ecuación, obtenemos la similitud con el usuario activo para todos los demás usuarios presentes en la base de datos, quedándonos con los k usuarios con mayor similitud. Un problema con el que nos podemos encontrar es que, si un usuario sólo tiene un ítem en común con el usuario activo, pero con idéntica valoración, la función de similitud devolverá 1, con lo que podría entrar este usuario en el vecindario. Esto no es realista, ya que solo tienen una película en común y eso no nos ayuda a la hora de obtener recomendaciones adecuadas. Para evitar esto, solo calculamos la similitud de los usuarios que tengan, al menos, 5 películas en común con el usuario activo.

Una vez tenemos definido el vecindario, buscamos los ítems a recomendar al usuario activo entre los que han sido valorados por sus vecinos y no por él. Para obtener estas recomendaciones, estimamos la valoración que le daría a cada ítem el usuario activo u usando la siguiente ecuación:

$$\hat{r}(u,i) = \bar{r}(u) + C \sum_v sim(u,v)(r(v,i) - \bar{r}(v)) \tag{3}$$

Donde la constante C se calcula de la siguiente manera:

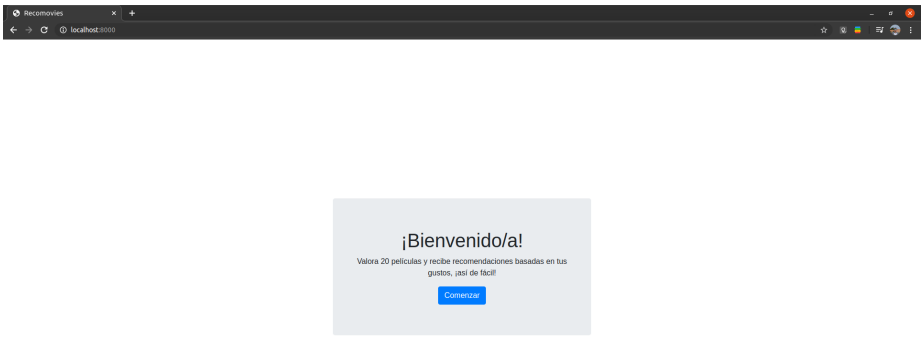
$$C = \frac{1}{\sum_v |sim(u,v)|} \tag{4}$$

Cuando ya tenemos la valoración estimada de todos los ítems, seleccionamos solo los de valoración más alta (4 o 5), que son las recomendaciones que le mostramos al usuario activo. No obstante, nos hemos encontrado con que, al sumar el valor $\bar{r}(u)$ en la fórmula de estimación de la valoración, obtenemos valores mayores de 5 para la misma. Para solucionar esto, lo que hacemos es reescalar todo el conjunto de valoraciones estimadas para que estén dentro del intervalo $[1, 5]$ usando la siguiente fórmula, donde x es la valoración estimada a reescalar y max es la valoración máxima que se ha obtenido.

$$\frac{x-1}{max-1}4 + 1 \tag{5}$$

3. Interfaz

En esta sección, vamos a mostrar la interfaz de usuario de la aplicación web que se ha desarrollado para servir como *frontend* al sistema de recomendación. Al acceder a la web, el usuario es recibido con un mensaje de bienvenida que le explica sencillamente en qué consiste el sistema.



Cuando el usuario pulsa en el botón de Çomenzar", se crea una nueva sesión para él y se obtienen de forma aleatoria las 20 películas que debe valorar. Estas películas se le muestran de forma secuencial al usuario en pantallas como la que se puede ver a continuación:

Una vez se han valorado las 20 películas, el sistema calcula el vecindario y obtiene las recomendaciones, mostrándoselas al usuario en la pantalla de resultados.

3.1. Manual de uso

Para ejecutar la aplicación, debe tener una carpeta *data* en la raíz del proyecto que contenga los ficheros *u.data* y *u.item* con los datos. La aplicación leerá estos datos y los cargará en una base de datos SQLite también en la raíz.

Debe tener instaladas en sus sistema las herramientas de Go, con lo que podría lanzar la aplicación con el siguiente comando:

```
go run main.go
```

En la terminal se le mostrará el progreso de carga de datos e inicialización del servidor web, tras lo cuál, debería poder acceder a la aplicación con su navegador web a través de la dirección *http://localhost:8000*.

4. Referencias

Para la realización de esta práctica se han usado algunas fuentes complementarias al guión proporcionado, las cuáles se indican a continuación:

- Documentación de Go, usada para consultar dudas sobre ordenación de *arrays*.
- Documentación de Gin, *framework* usado para el desarrollo de la aplicación web.
- Documentación de GORM, usada para consultar cómo se componían las consultas a la base de datos.